AD-A055 209 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 5/1
SIMULATION OF THE ACQUISITION MANAGEMENT INFORMATION SYSTEM.(U)
MAR 77 A H LINDER

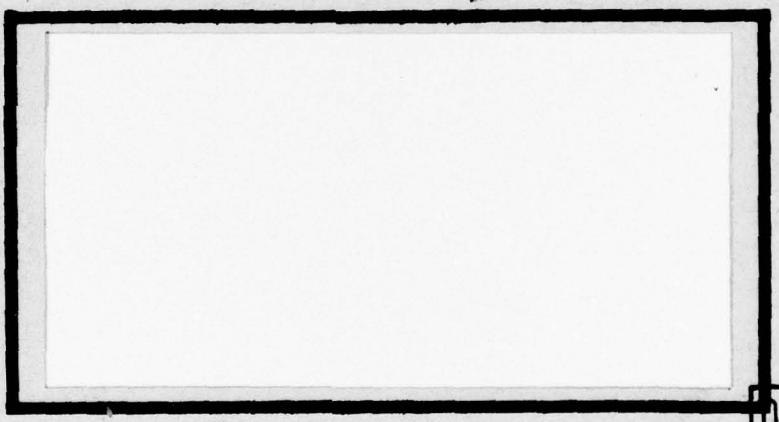
UNCLASSIFIED AFIT/GCS/MA/78-1 NL

1 OF 2
AD
A055209

AD A055209

FOR FURTHER TRAN

THIS DOCUMENT IS BEST QUALITY PRACTICABLE
THE COPY FURNISHED TO DDC CONTAINED A
SIGNIFICANT NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.



DDC
RECEIVED
JUN 19 1978
E

UNITED STATES AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

CIVIL ENGINEERING SCHOOL

AD No.
DDC FILE COPY

78 06 13 044

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

(1)

(6) SIMULATION OF THE ACQUISITION MANAGEMENT INFORMATION SYSTEM.

(9) Master's THESIS,

(11) Mar 77

(14) AFIT/GCS/MA/78-1

(10) Alfred H. Linder III
2Lt USAF

(12) 129 p.

DDC
RECEIVED
JUN 19 1978
E

Approved for public release; distribution unlimited

78 06 13 044

012 225

AFIT/GCS/MA/78-1

SIMULATION OF THE ACQUISITION MANAGEMENT INFORMATION SYSTEM

THESIS

Presented to the Faculty of the School of Engineering

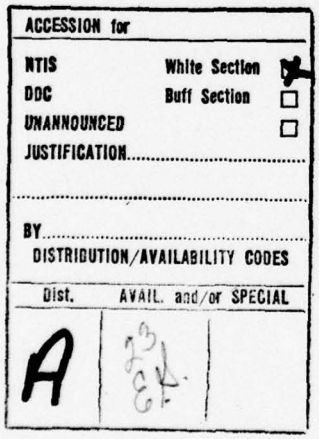
of the Air Force Institue of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science



by

Alfred H. Linder III, B.S.

2Lt USAF

Graduate Computer Science

March 1977

Approved for public release; distribution unlimited

PREFACE

This investigation is a result of an effort to provide AFSC, Wright-Patterson A.F.B., Ohio with a computer simulation analysis of the IBM 370/155 computer system's throughput performance and batch-interactive-mix. I hope that the results will prove useful to the Acquisition Management Information System (AMIS) department.

I wish to express my sincere thanks to Major Kenneth Melendez of AFIT/ENC for his advice and leadership as my advisor and his interest in this effort. I also want to thank my wife Linda for her emotional support throughout this undertaking.

Alfred H. Linder, III

CONTENTS

	Page
face	ii
t of Figures	v
t of Tables	vi
tract	vii
Introduction	1
Background	1
General Discussion	3
Problem Statement	4
Scope	4
Approach	5
IBM 370 Operating Characterisitcs	8
System Organization	8
Main Storage	9
Central Processing Unit	10
Input/Output	10
Byte Multiplexer Channels	11
Block Multiplexer Channels	11
Job Scheduler	12
Multiprogramming	14
System 2000	14
Data Bank	15
Data Base Management System	16
Data Dictionary/Directory System	16
Data Base Administrator	16
User System Interface	16
System Requirements, Definitions and Performance Measures	18
Job Processing Sequence	18
Workload Parameters	20
Performance Measures	22
Development of Simulation Model	26
Simulation Steps	28
Model Formulation	28
Logic Flow	31
Data Preparation	33
Programming Techniques	33
Validation of the Simscript Model	35
Assumptions	35
Steady State	36
	38
	40
	41
	43
	43
	46
	48
	49
	51
	66
	120

Preceding Page BLANK - NOT FILMED

LIST OF FIGURES

Figure		Page
1	Computer Response Time	2
2	IBM System/370 Logical Structures	8
3	Organization of Buffer Memory Hardware	9
4	Data Base Management: Conceptual Environment	15
5.	Simulation Process Flowchart	24
6	Job Processing Sequence	29
7	Transient Warm-up Period	37
8	Nomograph for Cost Estimates	41

LIST OF TABLES

Table		Page
I	Scheduling Priorities	13

ABSTRACT

The emphasis of this investigation is on the development of a simulation model that will allow an analysis to be performed on the batch-interactive-mix and throughput performance of the IBM 370/155 computer, used by the Acquisition Management Information System (AMIS) users. The model is driven from data obtained from the AMIS major production jobs and hence the validity of the results are not accurate. In the recommendation section of this effort, cluster analysis is presented as a method to collect data about all the types of AMIS jobs so that a follow on study could be made to achieve valid results from the simulation model that has been constructed.

The basics of hardware and software are discussed, along with some information concerning the System 2000 data base management software package. After the operating characteristics have been discussed, the need for system requirements, definitions and performance measures are presented and the necessary simulation steps are enumerated to show how the simulation model was constructed.

The main variables of interest (CPU utilization and gain factor) are analyzed to determine the throughput performance of the system as the interactive workload is increased and as the number of interactive ports are decreased.

SIMULATION OF THE ACQUISITION MANAGEMENT INFORMATION SYSTEM

I. Introduction

A time sharing system should provide the best possible throughput performance for batch and interactive jobs. Throughput is the amount of useful work completed per unit of time given a particular workload. (Ref. 30:16) Even after a computer system has been designed and implemented it may be necessary to perform a batch-interactive-mix analysis to determine if the system is being used in such a manner that high throughput performance is achieved. A computer simulation analysis of the operating system might reveal pertinent information to aid in the determination of the exact number of batch and interactive terminals to be used in achieving good throughput performance. A simulation model was developed to analyze the throughput performance for an Air Force System Command (AFSC) organization.

Background

The Air Force System Command (AFSC) at Wright Patterson Air Force Base uses the System 2000 software package to store contracts as they are developed from the conception phase through the completion phase, as well as during the purchasing phase of the acquired system. There are approximately 75 terminals, used at several locations across the United States. These are connected to the System 2000, which runs on an IBM 370/155 model computer.

The Acquisition Management Information System (AMIS) was implemented to support contract administration and disbursement activities. One of the major objectives is to implement Source Data Automation (SDA) at the buying activities, AF Plant Representative Offices (AFPROs), and the

Air Force Contract Management Department (AFCMD) - thus providing them with an interactive capability to update and query the central data base.

Throughput performance could possibly be improved if a batch-interactive-mix analysis is performed and the best ratio of batch to interactive terminals is used by AFSC. A simulation program wich models the System 2000, I/O channels, controllers, memory allocation technique of the IBM 370, job scheduling, and outputting of jobs would provide results which can be used to determine the best batch-interactive-mix. Good throughput performance assures that the work being processed will be completed in time for its intended use as illustrated in Figure 1.

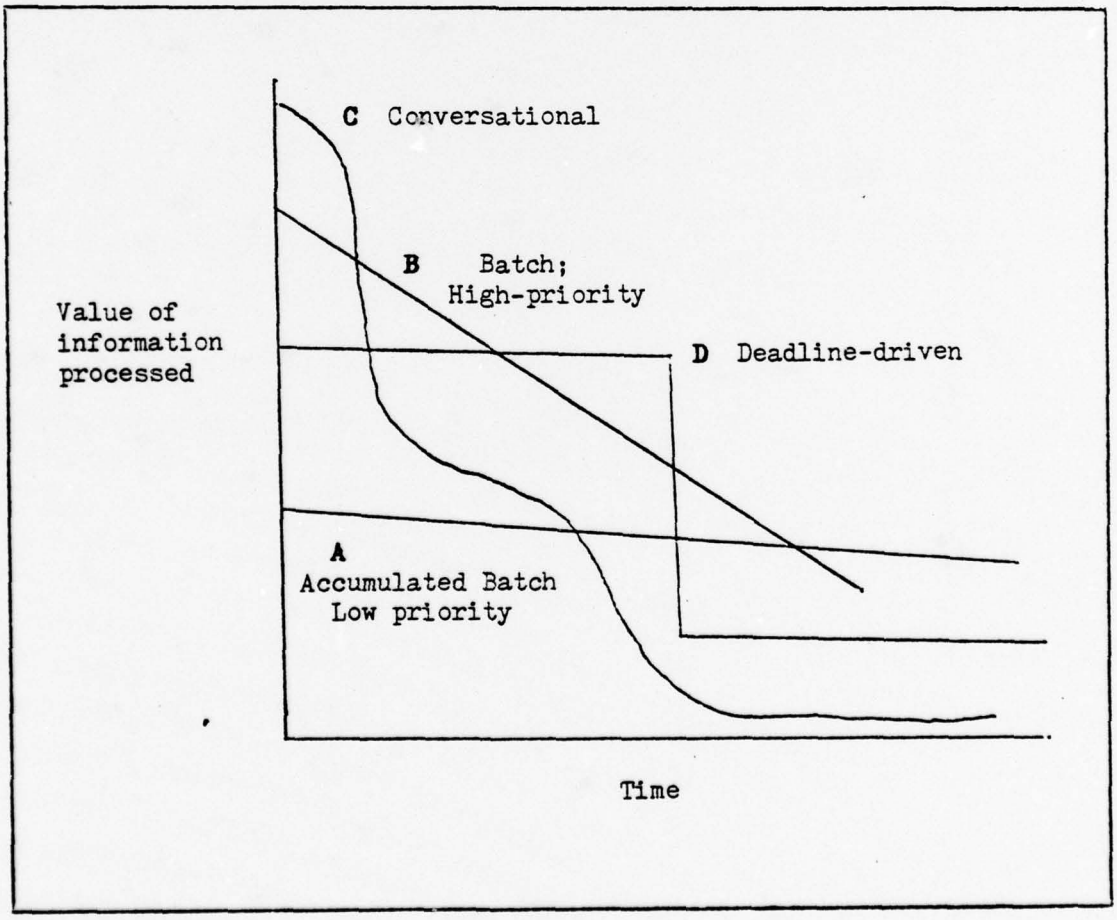


Fig 1. Computer Response Time (From Ref. 8:11)

If there is a deadline to getting a job completed and the throughput performance of the system is low then the job that needs to be executed may not be completed in time. Efficient use of computer terminals is necessary to minimize the expense of processing data and of retrieving meaningful results from the computer.

General Discussion

Batch and interative terminals have advantages and disadvantages and a batch-interactive-mix analysis based upon the types of jobs run by an organization will provide results that can be used to efficiently use computer terminal resources.

Terminals that provide access to and from the computer are usually either batch or interactive. The term batch implies that the entire program or data is routed to the computer concurrently and the results are routed back to the terminal after the complete execution of the program. Interactive processing allows small segments of data to be transmitted to the computer with results being sent back to the terminal before more data is required.

When a batch job is submitted through the terminal, the job will be scheduled for processing according to the job size, according to the availability of the needed peripheral processors, according to the job priority, and in accordance with any other constraints the operating system designers may have employed. These scheduling constraints can cause delays, especially if large jobs require a large section of central memory. Several small jobs can tie up most of the central memory available, and when one job finishes there still may not be enough central memory available for the extremely large job that has been waiting in the job queue. This will force the job scheduler to by-pass the larger job because a smaller job in the queue is available for processing. If the

large job has a high enough priority, some of the smaller jobs might be rolled out of central memory to allow the larger job enough memory to begin execution.

Scheduling batch jobs can become complex. "Its [scheduling system] objective is to select jobs from all jobs available to provide a well-balanced mix, thus enabling the computer to use its resources efficiently, and at the same time meet all its deadlines (Ref. 3:27)."

Use of interactive terminals can minimize these scheduling problems because their demands for processing is recognized almost instantaneously due to the higher priority of interactive jobs. However, this creates a new significant problem. "With interactive processing ... system resources must always be fully prepared to handle a worst case situation (i.e., amount of memory needed)." (Ref. 8:14)

A single computer can handle hundreds of terminal connections, and the number of batch terminals versus the number of interactive terminals can significantly alter the operational costs of using the computer. The approximate ratio of batch to interactive terminals that should be used is dependent upon the computer system and the needs of the organization the computer services.

Problem Statement

AFSC is interested in the results obtained from batch-interactive-mix analysis performed on the AMIS jobs run on the IBM 370. The results might aid in making operational changes that will increase the throughput performance for AMIS users.

Scope

The objective of this investigation is to build a computer simulation model of the IBM 370 and System 2000 which will allow a batch-interactive-mix analysis to be performed and the results presented to AFSC. The model

will be driven by estimates of the workload characteristics of the AMIS jobs. A section is presented dealing with obtaining valid input data which will lead to more accurate results from the simulation analysis.

Approach

One way to evaluate and determine if an organization is achieving near optimal throughput performance by using the best batch-interactive-mix, is to construct a computer model of the system being used and to use input data that is representative of jobs being run. When large samples of workload characteristic data is needed, a computer simulation model will help the analyst arrive at meaningful benchmarks. The sheer volume of data, needed to drive the simulation model, would rule out the use of achieving the analysis by use of a mathematical probabilistic model.

Simscript II.5 is a computer language that is ideal for simulating the computer operating system because it allows the user to schedule events when a pre-determined simulation time has elapsed. Results from the Simscript computer simulation run can be used to evaluate the performance of the system. System performance actually involves two primary considerations. The first is the effectiveness of how the system handles a specific job or request; the second involves efficiency or how the system uses the resources available. Understanding both of these considerations is essential when analyzing the performance of the system. The most efficient use of system resources may not provide the best throughput performance for a specific job, but it certainly will provide good throughput performance over a given time period, with a substantially reduced operational cost.

The computer simulation model must be capable of handling certain workload parameters such as: job CPU time, job I/O request, CPU service

time, I/O service time, interarrival time, priority, memory requests, number of simultaneous users, number of jobs in the system, etc. (Ref. 29: 12-13). Without understanding these workload parameters, or using them improperly in the simulation model, the performance evaluation results will be invalid or, at best, somewhat misleading. Workload characterization of jobs and tasks should be fairly accurate if the workload parameters are representative of the jobs run by the organization.

Workload parameters can be combined, consequently producing data that is representative of the jobs and tasks. Through a clustering analysis technique the analyst produces data that is consistent, regardless of the possible extreme values of a given parameter. According to Anderberg,

> "Cluster analysis has been employed as an effective tool in scientific inquiry. One of its most useful roles is to generate hypotheses about category structure. An algorithm can assemble observations into groups which prior misconceptions and ignorance would otherwise preclude. An algorithm can also apply a principle of grouping more consistently in a large problem than can a human. ... cluster analysis may be used to reveal structure and relations in the data. It is a tool of discovery. (Ref. 3:4)

Developing a simulation model which uses correct workload parameters and valid input data which allows the analyst to examine the systems performance is more than an intuitive art. Shannon [28] has suggested some criteria that are instrumental in developing a successful simulation model. The 11 suggested steps will help the simulation designer during the initial state of "system definition" through the "documentation" phase of the design. (Ref. 28:23) These steps will be discussed in detail in Chapter 3.

Understanding the system and analyzing the operation were important in building the Simscript II.5 model of the AMIS System. This enhances the development of a workload analysis methodology. Chapter 2 deals with the operating characteristics. Chapter 3 discusses the job processing

sequence and performance measures while Chapter 4 discusses the model formulation, logic flow and data preparation. The first phase of a performance improvement effort is that of understanding the computer operating system and the logical structure of the computer, which is the topic of the next chapter.

II IBM 370 Operating Characteristics

System Organization

> "The IBM system 370 model 155 is a high-performance data processing system that provides the reliability, availability, and convenience demanded by business and scientific users, as well as by users with applications in communications or control." (Ref. IBM System/370 System Summary: 6-9)

The following sections: system organization, system control and System 2000 will be described using information from the IBM System/370 System Summary, IBM System/370 Principles of Operation and System Operation and Guide to Data Base Mangement.

Figure 2 shows the logical structure of the model 155 system.

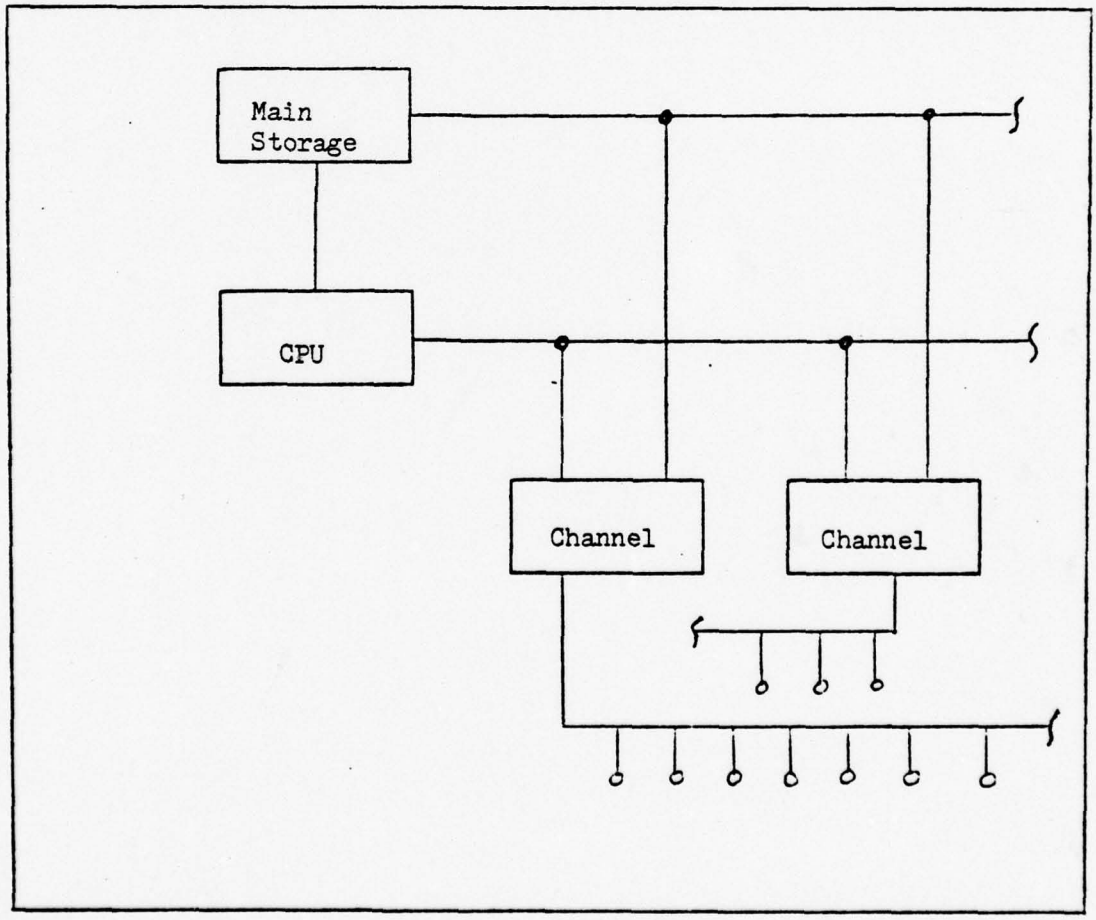


Fig 2. IBM System/370 Logical Structures (From Ref. 21:13)

The single CPU system logical structure is comprised of main storage, a central processing unit, a selector and multiplexer channels which communicate with each other by connecting paths.

Main Storage

Both data and program must be loaded into main storage before processing is allowed. The main storage is set up to provide direct addressable fast-access storage of data. The main storage is comprised of a large-volume access buffer called a cache.

In this type of a buffer storage system a Storage Control Unit (SCU) is placed between the processor and main storage unit and is illustrated in figure 3.

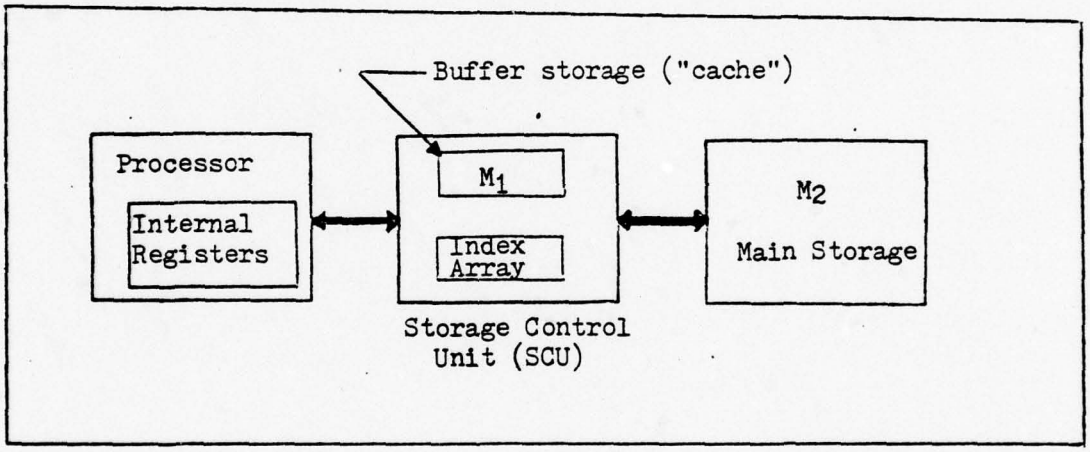


Fig 3. Organization of buffer memory hardware (From Ref. 14:192)

Exact copies of main memory (32 bytes at a time) are brought into the cache whenever a reference is made to an address that is not already existing in the cache. A block is stored over the existing block in the cache. Whenever a fetch request is generated, a check is performed to determine if there is need to bring in a different block from main memory. This algorithm is simple enough to be built into the hardware

of the system and allows a much faster fetch request from "copied" portions of main memory which actually reside in the buffer storage "cache" unit.

Central Processing Unit

The CPU is the controlling center of the computer system and handles system related functions such as execution, interrupts, initial program loading, etc. The CPU will handle 5 basic classes of instructions: system control, general decimal, floating point, and input/output instructions. The basic instruction sets include decimal, floating point, extended precision floating point, direct control, byte oriented operand, dynamic address translation, extended control mode, and store status and program reset.

Input/Output

Data is transferred between devices and main storage by attaching the I/O devices to channels, and the actual communication between control units and the specific channel takes place over a connector called the I/O interface. I/O devices such as card reader, punches, magnetic tape units, disc storage, drum storage, typewriter-keyboard devices, printers, teleprocessing devices and sensor-based equipment are handled by the associated I/O interface which provides information format and control signal sequences. I/O devices fall into several catagories, most of which are used for:

Auxiliary storage.

Machine and manual (keyed) input, both local and remote.

Teleprocessing.

Reading (or output) of external documents and displays.

process control.

and data acquisition.

"An Input/Output operation transfers data between main storage and an I/O device. An I/O operation is initiated by a program instruction that generates a command to a channel. A control unit receives the command via the I/O interface, decodes it, and starts the I/O device. (Ref. 21:2-7)

Byte Multiplexer Channels

Channels are the direct controllers of I/O devices and control units, and allow the system to read, write and compute. This is accomplished while relieving the CPU of having to communicate directly with the I/O devices. "The byte multiplexer channels separate the operations of high-speed devices from those of lower-speed devices."(Ref. System Summary: 2-7) This allows channel communication to take place in one of two different modes: the byte mode using the slower data rate I/O devices and the burst mode using the higher data rate I/O devices.

When operating in the byte mode, the single data path can be used by several low speed I/O devices (such as card readers, printers and terminals) and each device takes turn in sending data over the multiplexed line. This is accomplished when the channel receives and sends data to the I/O on demand and is controlled by the channel program.

In the burst mode, I/O devices (such as magnetic tape units, discs, or data cell storage) are not under the control of the programmer and after these high-speed devices have established a logical connection with a channel, large amounts of data can be transferred in "bursts" which are not multiplexed.

Block Multiplexer Channels

The block multiplexer channels operate high-speed I/O devices on a single data path and can also operate in one of two modes: block multiplex or selector mode. In the block multiplex mode the channels permit interleaving of channel programs and allow initiation and

termination of these programs to occur sooner than is allowed by selector channels. The block multiplexing mode allows more data to be transferred during its burst mode than can be routed by the byte burst mode. Entire records and blocks can be transferred during each burst so that block multiplex channels are used with faster I/O devices than are used by the byte multiplexor channels.

When operating in the selector mode, I/O devices are attached to the selector channel which transmits data to or from a single I/O device at a time. These select channels can be operated with either the slow or high speed devices but are especially suitable with high speed I/O devices. Once a selector channel has attached a particular I/O device it will transmit data until all data has been handled and no other I/O device can interfere with the selector channel.

Information about the hardware configuration has been presented and it is hoped that the reader is a little more familiar with the computer system. General information concerning the software of the system is discussed now to further aid in the understanding of the system.

Job Scheduler

All jobs are classified into 11 membership classes. The requirements for each class is listed in table 1. where K is the region size in memory (in blocks of 1024 bytes) and T is the CPU time in minutes. The job class represents the programmer's estimate of the resources required for the job and is used to schedule and prioritize jobs. The following classes are used on the IBM 370/155 by A.S.D.

Table 1.

Scheduling Priorities

Class	Region	Time	Tape Mounts Permitted	Disk Mount Permitted
A	K≤200	T≤5	NO	NO
B	K≤260	T≤60	NO	NO
and (K 200 or T 5), I.E., JOB CANNOT BE RUN IN CLASS = A				
C	260<K≤500	T≤60	NO	NO
D	K≤200	T≤5	YES	NO
E	K≤260	T≤60	YES	NO
and (K 200 or T 5), I.E., JOB CANNOT BE RUN IN CLASS = D				
F	260<K≤500	T≤60	YES	NO
G	K≤200	T≤5	YES	YES
H	K≤260	T≤60	YES	YES
and (K 200 or T 5), I.E., JOB CANNOT BE RUN IN CLASS = G				
I	260<K≤500	T≤60	YES	YES
J	K≤500	T≤60	YES	NO
L	K≤500	T≤60	YES	YES

Multiprogramming

Once a job has been assigned a class priority it is also ranked by priority within that class. The highest priority job will always have access to the CPU under this multiprogramming scheme. Once a job has accessed the CPU it will continue being processed until the job must relinquish the CPU because of an I/O or until a higher priority job has been scheduled or a higher priority job has completed its I/O. The job that was interrupted is put into a wait state and must compete once again with all jobs available to be processed in accordance with the priority established in Table 1. The amount of I/O a job performs does not affect the priority algorithm and this multiprogramming scheme works well if the higher priority jobs are I/O bound while the lower priority jobs are CPU bound.

Because there is no time slicing, the lower priority jobs must wait until all the higher priority jobs are handling I/O and this means that these jobs will be waiting for long periods of time in between opportunities to access the CPU. CPU utilization will usually be high with good turnaround time for the higher priority jobs. Degredation of turnaround time will occur for the lower priority jobs. Fortunately the AMIS jobs perform a lot of I/Os when run on the System 2000 data base management system. The large number of I/Os allow the lower priority jobs to frequently access the CPU.

System 2000

The system 2000 software package supports the AMIS requirement of storing and modifying contracts for Air Force Systems. This system simplifies the actual storing and accessing of contract related data.

"A data base is generally acknowledged to be a collection of multiple logical files containing interrelated but nonredundant data which can be accessed by one or more applications. A data base management system is a software tool -- actually a collection of routine -- used to define and maintain the data base's logical structure, and provide a means by which data can be retrieved." (Ref. 4:1)

The system 2000 data base is comprised of 5 functional elements: Data Bank, Data Dictionary/Directory System, Data Base Administrator, Data Base Management System and the User System Interface. The inter-relationship between these functional elements is shown in Figure 4.

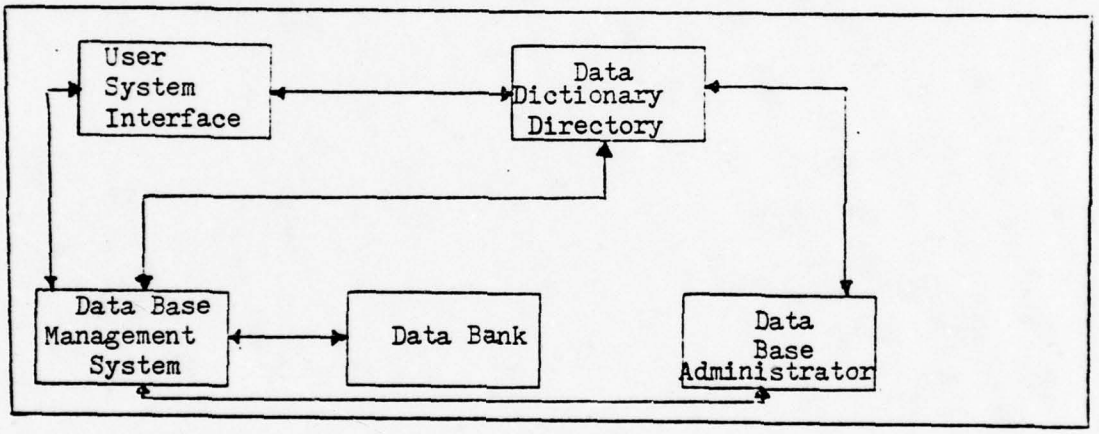


Fig 4. Data Base Management: Conceptual Environment (From Ref. 4:5)

Data Bank

The data bank is comprised of a collection of data bases organized to provide maximum performance of the system. The physical location of these data bases is not important as each is logically connected with the Data Dictionary/Directory System which is capable of coordinating centralized or decentralized data base files. Querries, updates, referencing, etc. can access the Data Bank through the Data Base

Management System.

Data Base Management System

The Data Base Management System consults the Data Dictionary/Directory System "... for information about data to provide the parameters necessary for the generalized software/hardware to execute user request." (Ref. 4:7) The Data Base Management System is the actual software package for accessing and storing data.

Data Dictionary/Directory System

The Data Dictionary/Directory System has two objectives. The first is the collection and dissemination of data which supplies the user with meta-data (data about data). The second function of the Data Dictionary/Directory System is that of establishing standards for coding conventions, data naming and usage.

Data Base Administrator

The major responsibilities of the Data Base Administrator (Ref. 4:6) are:

Definition of the content and structure of the data base.

Control of data access and modification rights to the data base.

Advising data base users on efficient techniques for extracting data.

Establishing data entry, edit, and validation standards.

Maintenence of the Data Dictionary/Directory System.

Maintenence of the Data Base Management System.

and keeping track of available physical storage.

User System Interface

To adequately serve its users the data base must use interfaces that allow for 3 main areas in the User System Interface Support.

The first is the language capability. The use of a natural language will help the user formulate requests and problem definitions that are easier to use when communicating with the system.

The second, "Interactive Capability -- the ability to browse in search of solutions supports the user decision process by providing the means to develop new alternatives." (Ref. 4:?) The last is the "Auxiliary Subsystems -- use will provide subsystems to assist users in putting system output into the most humanely comprehensible form. This category includes graphic techniques, algorithmic processes, and modeling and/or simulation tools." (Ref. 4:7)

The first phase in improving the performance of the computer system is that of that of understanding the system in terms of hardware configuration and related software programs in use. The second phase involves the analyzing of operations and understanding the system requirements, definitions and performance measures.

III System Requirements, Definitions and Performance Measures

Job Processing Sequence

The system simulation program that has been constructed deals with the following basic job processing sequence as outlined by MacDougall (Ref. 26:191-192). The simscript program itself is located in Appendix B.

1. Job arrival.
2. Request for central memory (CM) - if available, allocated, if not, the job is entered into the CM queue.
3. Request for central processor - if available, the job is assigned to it and executes until an I/O request is encountered or until execution completes or a higher priority job interrupts the CPU. If the central processor is not available then the job is entered into a CPU queue.
4. Request for an I/O - the job is released from the central processor and if another job is waiting for the CPU, the job waiting will be assigned to the CPU. If the disc is free, the disc is assigned to the job to process the I/O request; if the disc is busy, the request is entered in a queue.
5. On completion of processing of an I/O request, the disc is released and the central processor requested once again. (When the disc is released, the disc queue is checked; if there is a waiting request, it is assigned to the disc.)
6. When a job completes execution, it releases the central processor and its central memory space is released. (The CM queue is checked to

determine if there is a job waiting to which space now can be assigned, and the CPU queue checked to determine if there is a waiting job which now can be assigned to the central processor.)

7. The job leaves the system and the sequence is repeated.

The above processing sequence is modified slightly for interactive jobs because the interactive job can be thought of operating in either in one of two states. (Ref. 9:172) The think state is the time between a computer's response to the terminal request and when another request is made by the user, such as the hitting of the carriage return. The system state is the time from the user's request until the computer has processed that request. When the processed request has been completed an "interaction" has occurred and the interactive process has once again entered the think state.

Buzen [9] has verified that the average response time can be calculated by the following formula:

$$R = \frac{1}{J} \sum_{k=1}^{N} r(k) \qquad (1)$$

where,

" R = Average response time (i.e., average amount of time in system state per interaction).

r(k)= total time that the k-th interactive process (i.e., the interactive process associated with the k-th terminal) spends in system state during the observation interval (k=1,2,...N)."

J = Number of interactions completed during the observation time interval.

Buzen also claims that the average think time can be calculated by:

$$Z = \frac{1}{J'} \sum_{k=1}^{N} z(k) \qquad (2)$$

Where,

" Z = Average think time (i.e., average amount of time in think state per transition think state to system state).

J' = total number of transitions from think state to system state during the observation interval.

z(k) = total time that the k-th interactive process spends in think state during the observation interval (k=1,2,...N)." (Ref. 9:172-173)

Since this interactive process occurs, the simulation model must account for this response time, and during the system state points 3 through 5 of the job processing sequence justify considering the interactions as "separate jobs".

Workload Parameters

Within the basic job processing sequence the simulation program handles the following workload parameters as modified from the list compiled by Svobodova. (Ref. 29:12-13)

Job Cpu Time	Total Cpu time requested by the job or CPU time requested by a single interactive command.
Job I/O Requests	Total number of I/O requests needed by a single job.

CPU Service Time	Time required to process a single CPU operation.
I/O Service Time	Time to complete or process a single I/O task.
Interarrival Time	Time between two successive requests for any given resource.
Priority	Priority assigned to a job by the algorithm shown in Table 1.
Blocked Time	Time the job must wait for the CPU service in a wait state.
Memory Requests	Amount of core required by the individual job.
User Response Time	Time it takes the user to submit another request after a response from the CPU.
Number of Simultaneous users	The total number of interactive users concurrently logged on.
Number in the system	Total number of batch and interactive jobs operating within the system.

The Simscript programming language allows an entity to possess certain attributes which allow the simulation program to "move" a batch or interactive job "through" the system, carrying with it the necessary workload parameters to analyze the job workload characteristics. As the job moves through the system - enters queues, is assigned to the central processor, etc. - the job carries with it all the information needed throughout the simulation process. Once the "job output simulation"

occurs the individual job is destroyed in the system but the statistics about the job are kept by use of a TALLY STATEMENT. "The Tally Statement computes statistical quantities and prepares histograms for time-independent variables."(Ref. Simscript Manual) This allows the simulated job to be destroyed while specific data collection counters and routines to compute statistical quantities are used in conjunction with an ACCUMULATE statement to determine statistical analysis of the collective job workload characterizations. These statistics are useful in determining some important system performance measures, such as those listed by Svobodova. (Ref. 29:16-18)

Performance Measures

Throughput is a good measure of system responsiveness and since it is the amount of useful work completed within a given time period, given a particular workload, there are several performance measures that will give an indication of the system throughput performance. The turnaround time is the elapsed time between submitting a job and the time the results are received at the printer or terminal. The turnaround time can be a good indication of throughput performance. The elapsed time multiprogramming factor (ETMF) is a numerical value calculated by dividing the turnaround time of a particular job run under multiprogramming by the turnaround time of the job had this been the only job in the system. The ETMF is a measure of only one job in the system where gain factor is a measure of several sequential jobs.

The gain factor is determined by finding the time needed to execute a set of jobs under multiprogramming divided by the total system time needed to execute the same jobs sequentially without the capability of multiprogramming. Since types of jobs programmed may

vary considerably from day to day and during different time periods within a day, it is important that the gain factor be based upon results obtained from different time periods. Another measure of system performance is the CPU productivity or CPU utilization.

The CPU productivity is the percent of time the CPU is in use performing useful work. Multiprogramming should allow the system to perform with high CPU utilization because as one job has completed a CPU task and relinquished the CPU for an I/O there will usually be a job that has been waiting for the CPU in the CPU queue. Of course there is some overhead time, which is CPU time required by the operating system. CPU productivity could be very low if the organization jobs are consistently I/O bound.

The wait time for I/O is the time necessary to process an I/O task and therefore, if all the multiprogrammed jobs are involved in I/O tasks, the CPU will be idle until one of the jobs completes its I/O. If this occurrance of I/O bound jobs is common, there may be many times the CPU is idle and hence the CPU productivity will be low. If the CPU productivity is constantly low then the availability of the system will be high.

The availability of a system is the percentage of time the system is available to the users. Low availability will cause the external delay factor to be high (job turnaround time/the total CPU processing time required) and the throughput to be low.

A computer simulation model that uses good workload parameters and is capable of analyzing throughput performance with adequate performance measures will yield results that will help to remedy poor CPU utilization and improve the turnaround time for the organizational users. A reliable

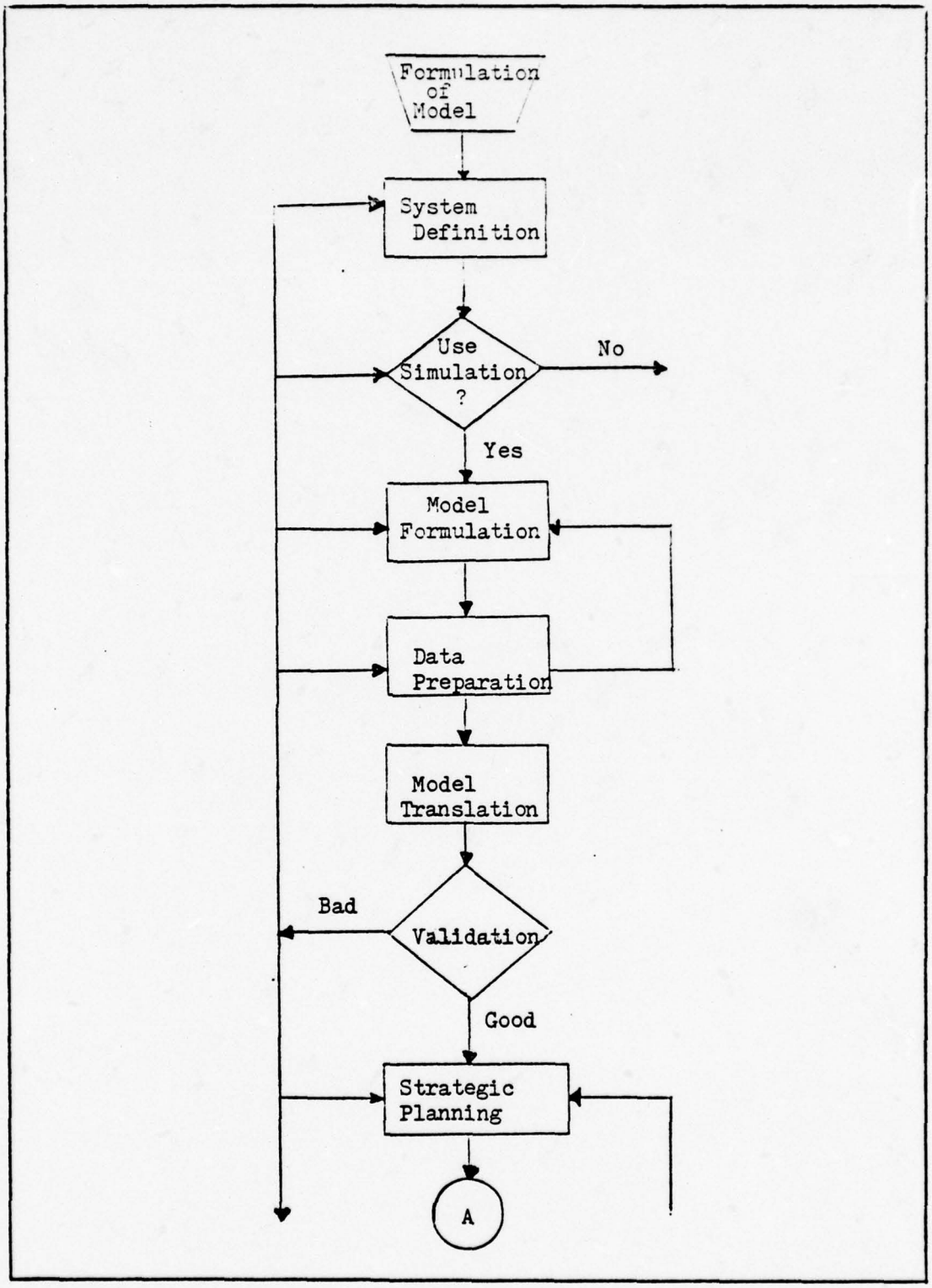


Fig 5A. Simulation Process Flowchart (From Ref. 28:24 Fig 1.3)

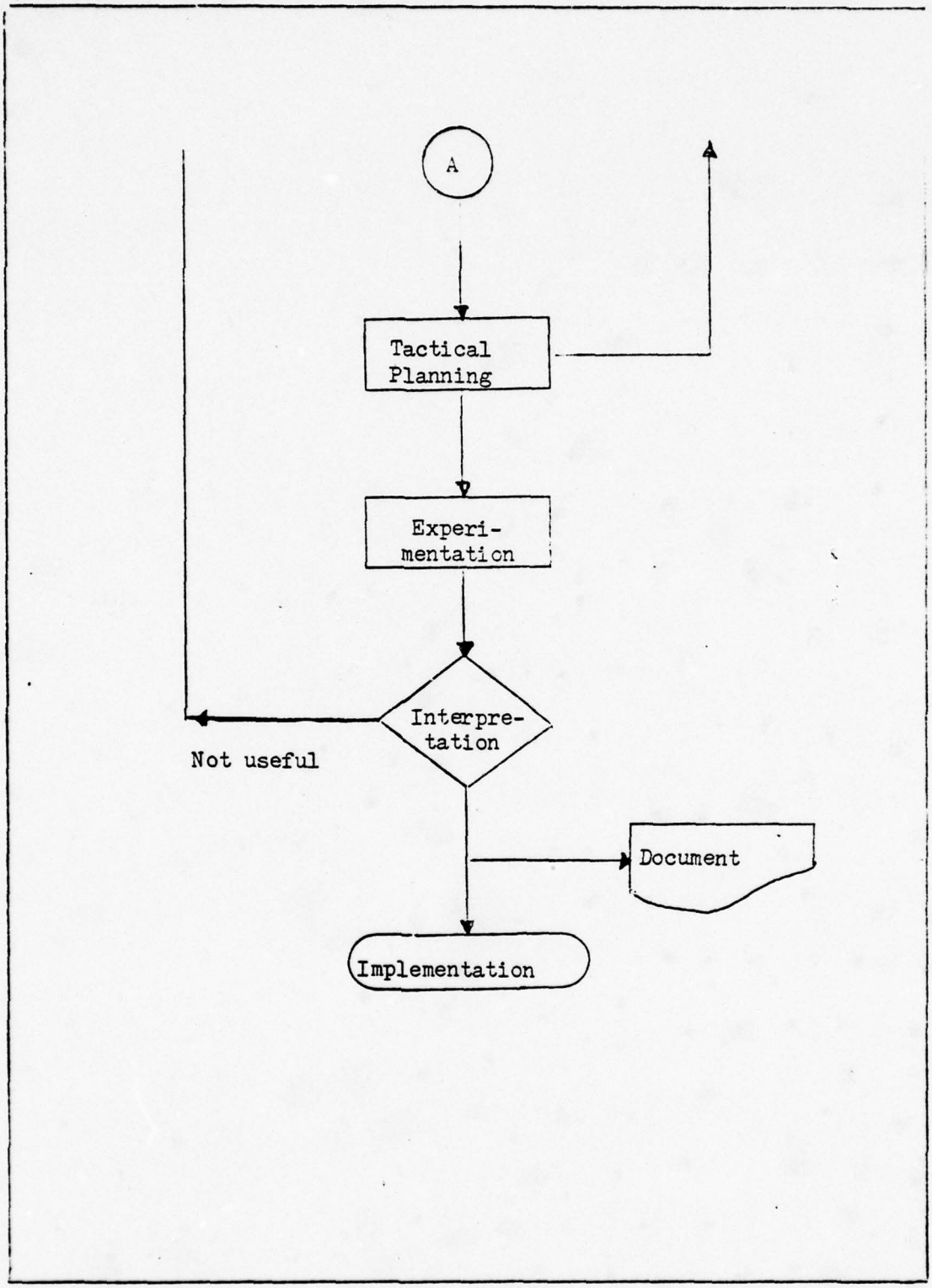

A
Tactical Planning
Experi-mentation
Interpre-tation
Not useful
Document
Implementation

Fig 5B

and accurate model is constructed by following the steps in the flowchart of Figure 5. (Ref. 28:24)

Development of Simulation Model

The flow chart on pages 24 and 25 shows the necessary stages that must be considered and implemented in order to achieve the construction of a reliable simulation model. The 11 steps that Shannon has incorporated into this process will be described in an abbreviated format. (Ref. 28:23)

1) System Definition - The Simscript II.5 model was developed within the boundary of considering the CPU time, I/O time, priority of the job and the need for disc or tapes. The measures of effectiveness of the model include: job turnaround time, CPU utilization, and gain factor.
2) Model Formulation - Reduction of the real system to be simulated into a logic flow diagram. This is shown in Figure 6A and 6B.
3) Data Preparation - Identification of data needed by the Simscript model which is described on page 33.
4) Model Translation - Description of the model in the Simscript language.
5) Validation - Increasing the level of confidence in the model and the results the model generates. This is discussed on page 35.
6) Strategic Planning - Design of an experiment that shows how the gain factor and CPU utilization varies with a change in the workload of the system.
7) Tactical Planning - Determining how the two experiments are to be run and tested.
8) Experimentation - Execution of the simulation model to perform these experiments and to perform a sensitivity analysis of the system. The sensitivity of the model is presented on page 38 and a

discussion of the experiments is located on page 40.

9) Interpretation - Drawing inferences from the data gathered from the experimentations. This will be done in the conclusion section of this paper.

10) Implementation - The results of these experimentations will be presented to AFSC.

11) Documentation - The results of the experimentations are listed in Appendix A.

The formulation of the problem has been presented in the Introduction while the system definition was presented in Chapter 3. It is now time to consider the use of a simulation language and to look at what is involved in the Model Formulation stage of the simulation process.

IV Simulation Steps

Model Formulation

Having discussed the problem to be investigated and showing the need for a computer simulation analysis, Shannon (Ref. 28:24) suggests that the next step concerns the reduction of the real system to a logic flow diagram that accurately depicts the process to be simulated. This flow approach to analyzing the IBM 370/155 will allow the workload parameters and performance measures to be incorporated into the simulation model. Figure 6 on pages 29 and 30 is a flow diagram of the necessary events to simulate the job processing of batch and interactive jobs.

Once the real system is represented by logic flow, the necessary data input and starting conditions are determined. Some data generation is internal to the program such as pseudorandom numbers and stochastic variates and must be generated appropriately. Once it is known how the real system should be represented, it is important that a simulation language be chosen that will allow a meaningful discription to be implemented in a computer simulation program. Unfortunately, the best simulation language for a particular investigation is overlooked simply because the analyst may be familiar with a specific language already and may feel that too much time and effort would be needed in determining which language is best and then having to learn that new language.

Heidenreich and Blitt (Ref. 7:38) have listed the advantages and disadvantages of different languages by comparing 9 languages including Simscript and GPSS simulation languages. Their comparison

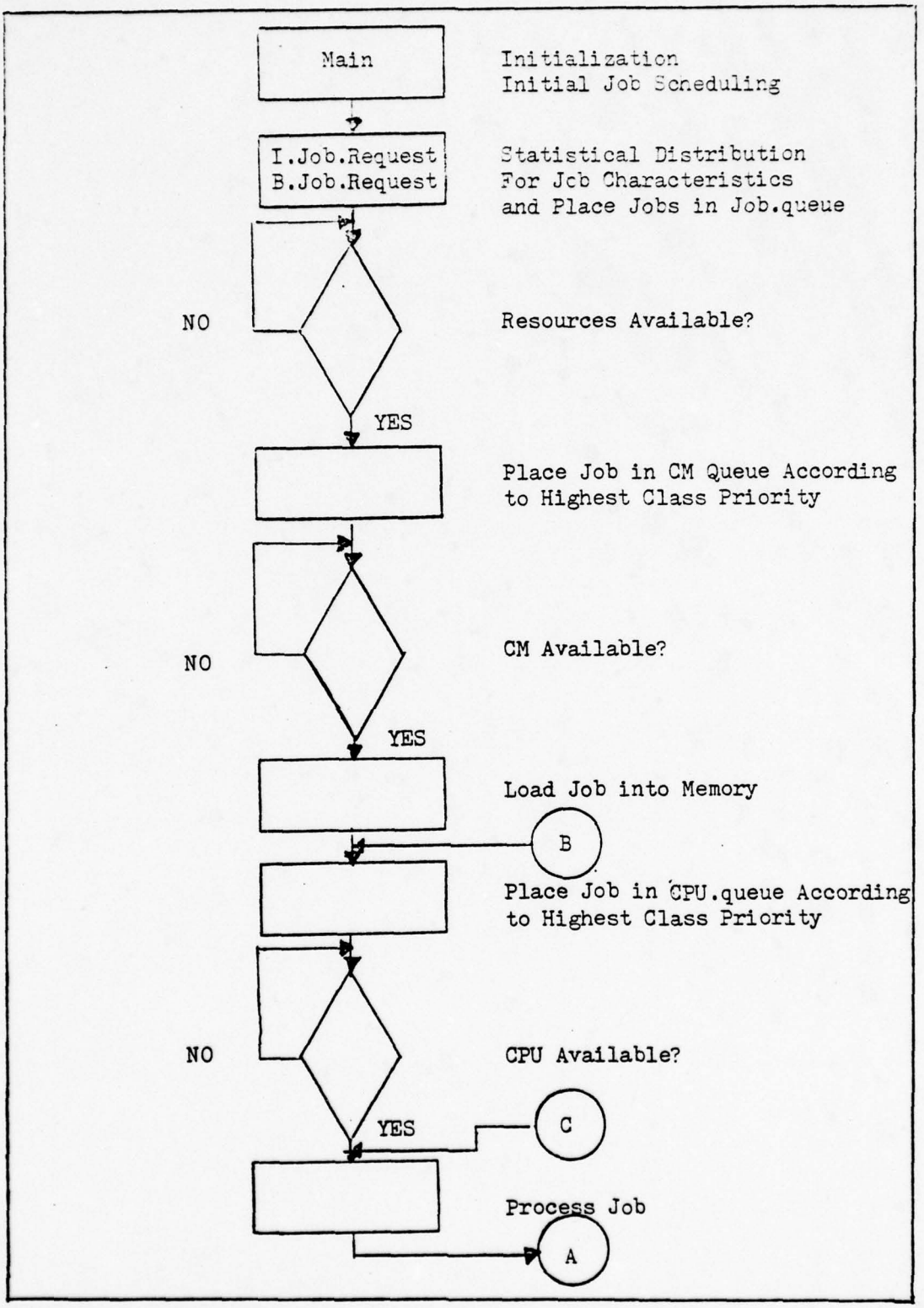


Fig 6A. Job Processing Sequence

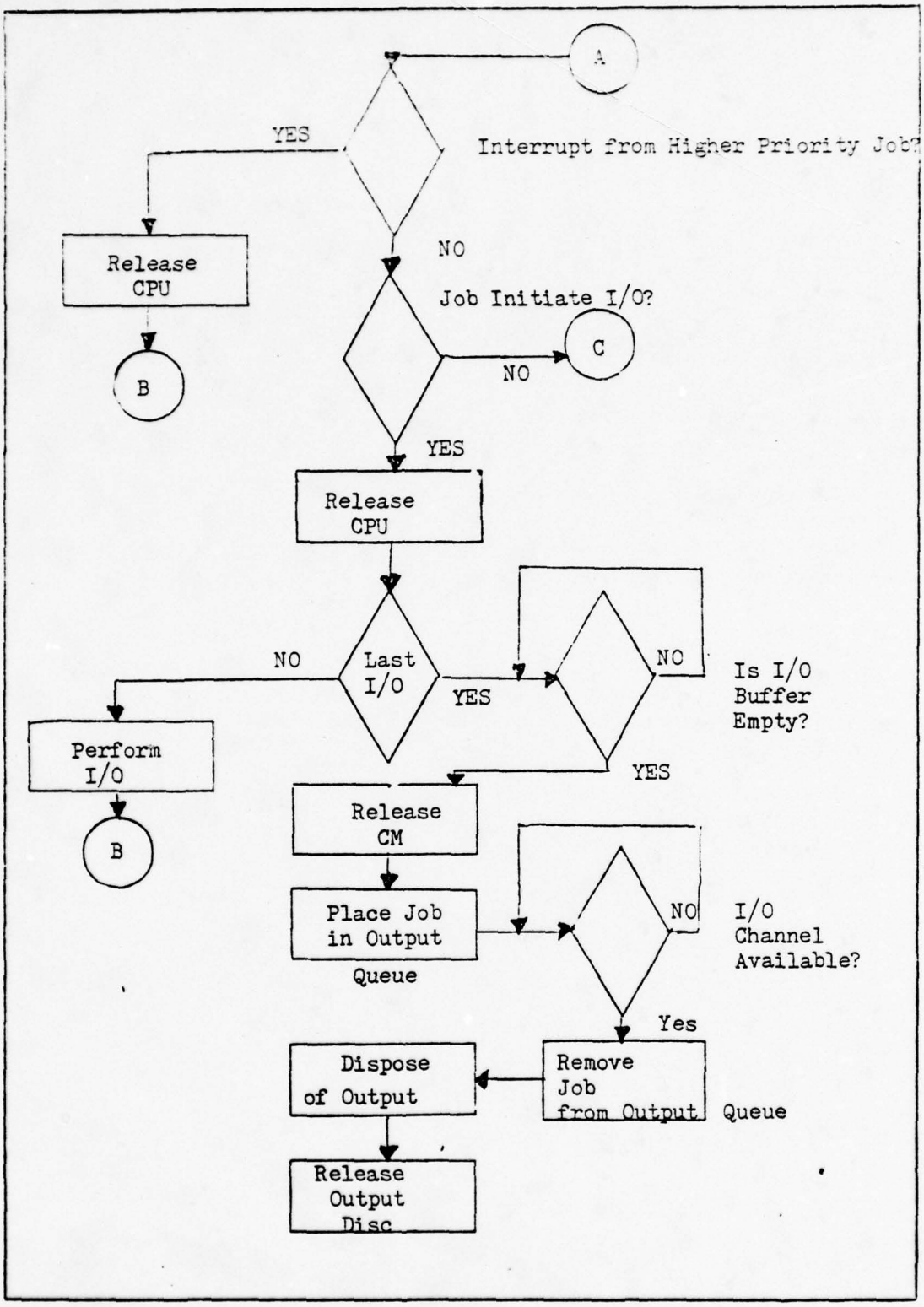

A
YES
Interrupt from Higher Priority Job?
Release CPU
B
NO
Job Initiate I/O?
C
NO
YES
Release CPU
NO
Last I/O
YES
NO
Is I/O Buffer Empty?
Perform I/O
B
YES
Release CM
Place Job in Output Queue
NO
I/O Channel Available?
Yes
Dispose of Output
Remove Job from Output Queue
Release Output Disc

Fig 6B.

suggests that Simscript is designed for simulation because it is event oriented but is slow to execute and not well known by its users. GPSS is also a simulation language but appears to be easier to learn than Simscript. A prime consideration between the languages might be simply "Which one is available to the investigator?". Simscript was choosen because it is suited to general programming and discrete-event simulation modeling, and because of familiarity with the language.

Logic Flow

Again, figure 6. shows the logic flow for the continuous simulation model which represents the AMIS system employed on the IBN 370/155. The actual program is listed in Appendix B. and a description of the logic flow will coincide with the events in the program model.

The "Main" routine is used to establish initialization of variables and to schedule the first interactive job and the first 6 batch requests. All simulation time is based upon parts of a Day, so all units must be established here in "Main" to avoid errors that result from working with wrong units. Once the variables have been initialized and all events scheduled, the "... 'Start Simulation' statement begins the simulation by passing control to the timing routine which removes the first event notice from the event set and executes that event." (Ref. 28:210)

Next, statistical generation of job input parameters is accomplished in the I.JOB.REQUEST (Interactive job request) and B.JOB.REQUEST (Batch job request) events. The individual job CPU time, I/O time, number of I/O's per job and carriage return, core requirements and job priority are generated within these two events. The job priority is determined by the algorithm shown in Table 1. and is based soley upon the amount of core requested, CPU time, and tape mount and disc mount requests. Once

the job characteristic parameters and job priority have been determined the 'job is placed into the job.queue'. The individual "job" entity has certain job attributes (job parameter values listed in the preamble) which are "moved" with the job throughout the simulation process by use of pointers when referencing the job. Once the job has been placed into the job queue the job scheduler scans the jobs in the queue to determine which job will get access to the central memory.

> "... the job scheduler chooses a small subset of the jobs submitted and lets them 'into' the system. That is, the job scheduler creates processes for these jobs and assigns the processes some resources. It must decide, for example, which two or three jobs of the 100 submitted will have any resources assigned to them. The process scheduler decides which of the processes within this subset will be assigned a processor, at what time, and for how long." (Ref. 14:211)

This processing management checks to see if necessary devices are available, creates processes for the job and then assigns memory when possible. The request for memory is based upon the job class priority (and the priority within a class), and when memory is available the job is loaded into core to wait for access to the CPU.

The job is now in the CPU.queue; this queue being formed every time the CPU is not available for job processing. When the CPU is released a job is selected from the CPU.queue according to the highest class priority and scheduled for the event "CPU.processing".

Once the job is processing, the job may be pre-empted by an interrupt that is initiated by a job which has a higher priority. At this point the CPU is released and the current job being processed is placed back into the "CPU.queue" and the higher priority job is scheduled for the event "CPU.processing". If the job being processed initiates an I/O, the job will release the CPU and the I/O task completed before the job is re-inserted into the CPU.queue (unless of course this is the job's last I/O).

If the job has completed its last I/O then as soon as the I/O buffer is empty, the core relinquished by the job may be used by another job waiting in the CM queue. When the I/O channel becomes free, the job will be placed in an output queue and appropriately disposed to the output terminal. After the job has been disposed to the output terminal the output disc is released for use by other jobs.

The event AN.DISC.RELEASE "destroys" the individual batch or interactive job but tallies up statistics about the job before removing the job's attributes from the simulation program.

Data Preparation

In order to simulate the job processing accurately there are five major parameters that must be generated for each job; namely, the amount of I/O time, the need for discs, the need for tapes, the amount of core needed and the amount of CPU time to process the job. To insure that sample jobs with these characteristics are selected and truly represent the job workload of the system would require an additional effort. This additional effort would require more time than has been allocated for this study. It was decided that the average statistics of S2K jobs (AMIS major production jobs, which account for about 25% of the workload) would be adequate to drive this simulation model. In the recommendations section of this paper the use of cluster analysis will be presented as a means for acquiring sample inputs which would reduce the amount of data that would need to be read into the program during execution.

Programming Techniques

The discrete-event capability of Simscript II.5 allows for good modularity throughout the simulation program. Modularization allowed the program to be divided into subprograms (events) which are called when the

timing routine has been scheduled. With this type of event, the need for global flags was minimized and consequently the hierarchical structure was forced to simulate the real system more realistically. Without the need for global flags the modifiability of the model is enhanced as well.

Modifiability implies that when changes are made to a portion of the program the changes in one subroutine make few, if any, changes in other subroutines throughout the program. This modifiability principle was used in constructing the two events which "parameterize" the batch and interactive jobs.

The internal generation of job attributes can easily be changed to read data as is recommended in the last chapter of this paper. This simplifies the transition process of modifying the program. Another way to make changes easy is to use the principle of understandability.

Simscript II.5 allows variables, routines and events to be in alphanumeric form and hence the principle of understandability is strengthened throughout the program. As long as the first 5 characters differ, there is no confusion when transferring execution from one event to another event or routine. This means that the names of the labels, events, routines and variables can be spelled out in a string of short works separated by "periods" when coding the program. Understandability is not merely a property of legibility as the entire conceptual structure of the model is involved.

The principles of modularity, modifiability, and understandability were achieved by using top down design methods in building the conceptual structure of this model. After the model had been built and debugged it was necessary to validate the model.

Validation of the Simscript Model

The investigator must have confidence in inferences and results obtained from the computer simulation results. Confidence in the model is established if the results from the model compare favorably and accurately with the real system results. If the model results vary drastically from those of the real system then changes must be made in the structure of the model or in the variables and parameter estimates that were used in the data preparation stage. If the model is simple to understand and easy for the user to control and manipulate then the analyst will find this iterative process manageable.

Before inferences can be drawn from the validated model, certain assumptions must be made. The input parameters must be as accurate as possible before a high level of confidence in the simulation results is achieved.

Assumptions

The AMIS jobs account for 90% of the workload on the IBM 370/155 and during a given typical week of November 1977, this would amount to about 12-15 hundred jobs. During this period the S2K jobs (one type of AMIS major production jobs) accounted for about one quarter of the jobs run. Data about these jobs revealed that of 266 interactive jobs an "average" job initiated 2,739 I/O's needing about 44 CPU-seconds for execution of the job. The typical batch job initiated 1,215 I/O's with about 34 CPU-seconds required. Since I/O time is not needed to determine class priorities it is difficult to ascertain close figures for job I/O times. It is known that the AMIS jobs are highly I/O bound jobs because of the time needed to search the data bases, so a ratio of 20 to 1 was chosen to represent I/O vs CPU time for batch jobs, and a

ratio of 4.5 to 1 for interactive jobs was selected. Assuming that the S2K jobs are "representative" of the workload, a 4 hour simulation time period was choosen to gather data for analysis. Again, the time allocated for this study did not allow for an analysis of data beyond this single 4 hour period.

Assuming that 28 batch jobs are scheduled during this 4 hour period and that 40 interactive jobs are run during this same time, the initial base line run showed that the model reflected a CPU utilization of 87% with a gain factor of 2.3684. (See definition of gain factor on page 22) This compares favorably with the 85% CPU utilization of the real system. Appendix A contains information about the baseline run and all other subsequent runs as well. The results were calculated after a warm-up period that is explained and discussed next.

Steady State

Shannon [28] recommends three possible ways to eliminate errors from simulation results that were introduced early in the warm-up of the system simulation.

1) Run the computer simulation long enough and the initial warm-up errors will be absorbed into an accurate average of the collected statistics.
2) Choose initial starting conditions that accurately reflect a given operating time period which will reduce errors during this transient period.
3) Throw out or just don't calculate statistics during an appropriate warm-up period.

The method employed in this investigation merged ideas from both

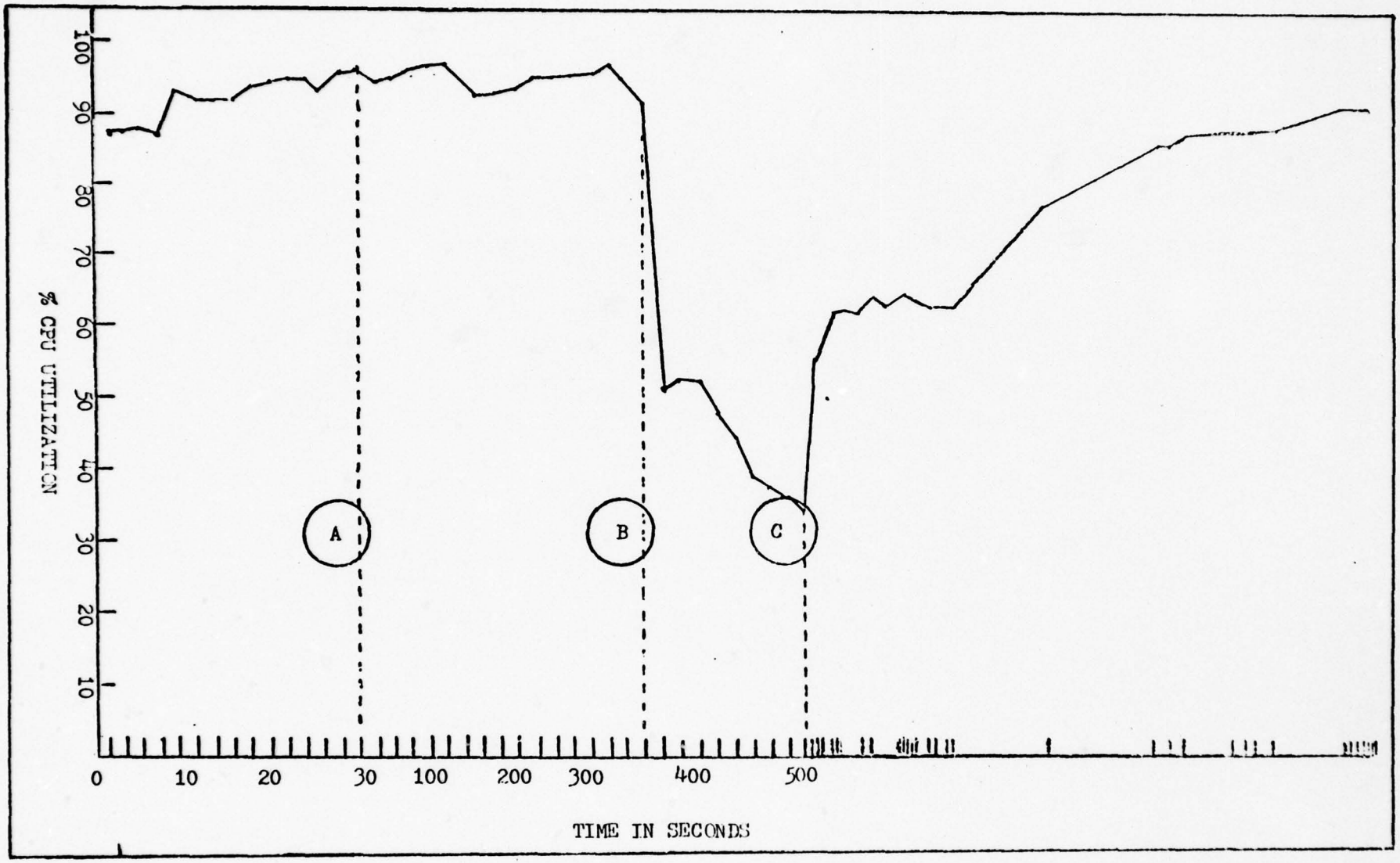


Fig 7. Transient Warm-up Period

points 2 and 3 above. The initial starting conditions provided for the scheduling of 1 interactive and 6 batch jobs to begin execution within the first minute of simulation time. After about 5 3/4 minutes a flag was set to start gathering job workload statistics (Figure 7 on page 37). shows what effect this warm-up period had on CPU utilization of the system. The time interval between "start simulation" and time A represents 30 seconds, while between point A and C each division represents a 20 second time interval. The cumulative CPU utilization remains fairly constant up to point B (approximately 5 3/4 minutes) when a flag was set to recalculate the cumulative CPU utilization. From point B to point C it can be seen that larger variations occurred. Point C represents that last regular interval where the CPU utilization was calculated and every point beyond C reveals the cumulative CPU utilization when any given job was completed. It appears that the cumulative CPU utilization stabilizes in the neighborhood of about 82 to 88 percent. By using point B as the starting reference point the results are weighted negatively by the rapid drop from point B to point C; but the immediate rise following point C will help keep this negative weight factor from drastically altering the true cumulative CPU utilization as it stabilizes near the end of the 4 hour run. Now that the transient warm-up errors have been reduced, a sensitivity analysis of the model's predictions is appropriate.

Sensitivity Analysis

A total of 29 simulation runs were made to determine what effect changes in I/O and CPU time would have on the cumulative CPU utilization and gain factor. The first graph on page 52 shows how a workload increase will affect the two variables. In this graph, as well as the rest of the graphs in Appendix A, the vertical dashed line represents the initial base

line run. It appears that the gain factor is on a steady increase and that CPU utilization falls below the baseline value of 86.67% after an increase of 10% in CPU and I/O time.

The chart on page 53 shows what effect an increase in I/O and decrease in CPU time will have on the variables of interest. The CPU utilization is on a "saw tooth" decline while the gain factor is also on a "saw tooth" rise which seems to rise when the CPU utilization decreases and falls when the CPU utilization increases. The baseline run is at a relative maximum point for CPU utilization with a slightly lower than average point calculated for the gain factor.

The chart on page 54 shows what effect changes in CPU time will have on the results when the I/O time is kept constant. The CPU utilization is on a constant rise until the CPU time exceeds the neighborhood of a plus 10% increase, at which point the CPU utilization decreases by about 5%. The baseline run reveals that a 10% increase in the CPU time of all jobs run will allow a 3% increase in CPU utilization while decreasing the gain factor by a factor of .2927.

The results on page 55 show that when keeping the CPU time (of all the jobs run) constant, an increasing change in I/O time will cause a gradual decrease in CPU utilization until going beyond a 10% increase, where this variable drops sharply. The gain factor remains on a gradual increase throughout the increase in I/O time.

The next eight charts, page 56 through page 63 show variations of changing the CPU and I/O times of all the jobs run. These graphs reveal that the baseline run is slightly below or generally above the CPU utilization figures and that the gain factor is at about the mid-point or generally below the values calculated for the other runs. These sensitivity runs

show that there are only two main areas where the results vary drastically from the baseline run. The first is where the CPU time is increased by 10% and the I/O time decreased by 10%; the other is where the CPU time is decreased by 20% and the I/O time is increased by 20%. Using the workload characteristics of the baseline run 10 more computer runs were made to help in the analysis of the batch-interactive-mix problem.

Experimentation

Two separate single factor experiments were run to determine what effect an increase in the interactive workload and a decrease in the number of ports available to run interactive jobs would have on the CPU utilization and gain factor. Five runs were made for each investigation with the intent that each separate run be considered a datum point.

The chart on page 64 shows that as the number of interactive jobs increase up to an additional 50% increase in the interactive workload, the CPU utilization climbs by about 8 1/2% while the gain factor decreases by a factor of .9550. There is every indication that the CPU could be used more efficiently but at the same time it appears that a decline in throughput and turnaround performance might result. The results on page 65 (decrease in the number of interactive terminals) are just about the same as those obtained when increasing the interactive workload.

Allowing just one factor (i.e., number of interactive terminals or interactive workload) to vary with each computer run provides a limited analysis concerning throughput performance. Ideally interactions between these two factors should be considered but due to the cost of each computer run it was decided that these extra runs would not be made. As an aid to the reader, the following discussion will help in the determination of the

important factors and the number of runs to be made during a simulation analysis.

Design Tradeoffs

Shannon [28], (Section 4.6) discusses a tradeoff study between the number of factors (input parameters, or variables), number of factor levels, number of replications and total number of computer runs required. Figure 8 shows a nomograph and the dark arrows show how to enter the nomograph and calculate the expected total computer cost and number of computer runs to be made.

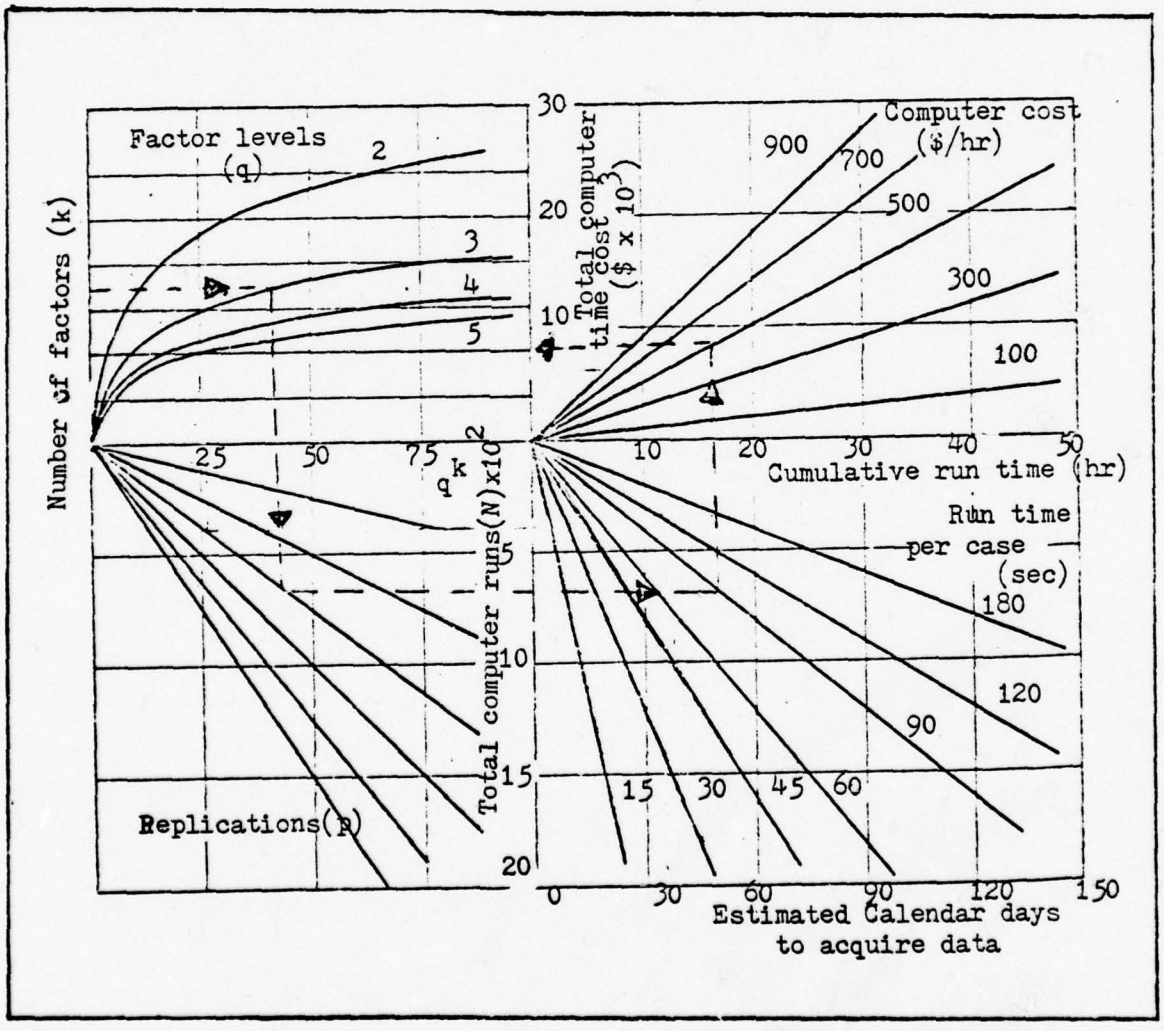


Fig 8. Nomograph for cost estimates (From Ref 28:156 Figure 4.2)

Where

k = number of factors (input parameters or variables)

q = number of factor levels

p = number of replications

N = total number of computer runs required

In section 4.6, Shannon lists 3 equations for determining which of the variables (level, factor and replication) are the most dominant. By knowing which variable is dominant the investigator can save computer costs by decreasing the level (if applicable - without altering results drastically) of the dominant variable. For example, if the number of factors is unquestionably the dominant variable then the number of factors might be reduced to save computer runs without sacrificing the importance of the results.

The "Pareto Principle" allows the investigator to reduce the number of factors because this principle revolves around the thesis that most systems need roughly 20% of the factors to explain nearly 80% of the simulation results, and that the remaining 80% of the factors account for only 20% of the observed performance of the system. (Ref. 28:153) The difficult part of reducing the number of factors is to actually determine which factors can be placed in the 20% high performance category.

The two main factors in the AMIS simulation are the amount of CPU and I/O time, and the response variables of experimental interest are the gain factor and the CPU utilization. These response variables allow an analysis of the throughput performance to be conducted, but as was mentioned previously, much more needs to be done in determining accurate input data which means that the CPU and I/O factors will produce more reliable results.

V Recommendations

The results from this simulation effort are not extremely accurate and could be improved by the use of cluster analysis. The input data, which currently is made up of about 25% of the workload, could become more representative of all types of AMIS jobs if cluster analysis were to be employed in gathering data.

Cluster Analysis

Cluster analysis is a statistical tool for analyzing data by developing a data classification or identification. The end result of this classification or identification is a collection of data that has a high "natural association" among members of the same group, and a much lesser association between data that has been clustered into different groups. There are several clustering strategies but all methods involve two primary considerations. "First, there is defined a measure of group-density or of inter-group likeness. Examples of the latter type of measure (The so-called 'similarity coefficients')....Secondly, the chosen measure has to be incorporated into a "sorting" strategy" whereby groups of elements are extracted." (Ref. 3:373) Anderberg (3) briefly discusses four metrics that might be used for clustering data which include the so-called "city block" metric, the Chebychev metric and the familiar Euclidean distance metric.

Euclidean Metric

Anderberg states that any metric must satisfy the following conditions:

1) $D(X,Y) = 0$ if and only if $X = y$
2) $D(X,Y) \geq 0$ for all X and Y in E,
3) $D(X,Y) = D(Y,X)$ for all X and Y in E,

4) $D(X,Y) \leq D(X,Z) + D(Y,Z)$ for all X, Y, and X in E.

where E is the symbolic representation for a measurement space and X, Y and Z are any three points in the space E.

The distance between data units can be found by using the Euclidean metric where

$$D_2(x_j,x_k) = \left[\sum_{i=1}^{i=n} (x_{ij}-x_{ik})^2\right]^{1/2} \tag{3}$$

When dealing with two or more variables, the Euclidean metric can be used to determine clusters and Green (18) has suggested the following steps to compute the cluster analysis.

1. Each variable (characteristic) is transformed so that the data becomes a standard distribution where the following rule converts the variables to a standardized variate with zero mean and standard deviation.

$$z = \frac{x - u}{\sigma} \tag{4}$$

2. Distances between all possible pairs of data are calculated using the Euclidean Metric.

3. The pair of data points with the smallest distance between the two points is chosen as the initial node of the first cluster and the

centroid of this pair is calculated.

4. "Additional points are added to this cluster (based on "closeness" to the last-computed average) until:

   A. Some pre-specified number of points has been clustered, or

   B. The point to be added to the cluster exceeds some pre-specified distance cutoff number." (Ref. 18:391)

Anderberg suggests using the single-linkage method for accomplishing this step in the cluster analysis. As new data points are added to the cluster the new centroid must be calculated so point B is not violated when an additional point is merged or "linked" to the cluster. If it is known that a certain distance between the cluster centroid and the nearest neighbor can not exceed some "coarsening parameter" value, then a new cluster node is established by determining another centroid between the points with the smallest distance between them.

5. The new additional cluster node is used as a basis for building a new cluster and step 4 is repeated.

6. To avoid artifically fine distinctions between clusters, the points may be allowed to be in more than one cluster.

The above has been a general discussion of cluster analysis; the section that follows will suggest a practical use of this technique to refine the data generation employed to drive the simscript model.

Practical Use of Cluster Analysis

A computer program which performs a cluster analysis on the input data (i.e., CPU time, amount of core needed, and need for tape or disc mounts) should be the next step in the analysis of the system. Forgy (31) has provided a simple algorithm consisting of the following steps, which will help in such a cluster analysis:

1) Since there are 11 general priority class memberships, the end points, for both CPU time and core needed, within each class should become the initial cluster boundaries.
2) Using the two end points in conjunction with the mid-point of the boundaries as seed points, run the data to allocate each data unit to the cluster with the nearest seed point. The seed points must remain fixed during the complete run.
3) Compute new seedpoints from the centroids of the resulting clusters.
4) Re-run the data using the new centroids as seed points and repeat this iteration until no data units change their cluster membership.

It is not certain just how many runs will have to be made before this iterative process stabilizes, but Forgy suggests that from empirical evidence, this will ordinarily be accomplished within the first 5 runs.

When the cluster analysis has been completed there should be a total of 33 separate clusters, each having a centroid value for the CPU time and core needed. The frequency of each cluster (the number of data points within each cluster) should be calculated and the data is then ready to drive the simulation model.

Now that the data has been reduced to 33 clusters that accurately represent the workload characteristics of the system, the "centroid data"

can be called at random according to the frequency of each cluster. This means that 33 data points, called at random, will suffice for valid inputs. This is much superior to the internal generation of values, the method employed to this point, and allows the analyst to include data from all the different types of jobs run on the system.

VI Conclusions

1) Using the data about S2K major production jobs, the baseline simulation run generated a CPU utilization of 86.67%. This value is close to the real system CPU utilization of 85% but it must be remembered that the results in this investigation were not obtained from using data established by the cluster analysis technique in the previous section. In most of the charts in Appendix A the baseline figure for the CPU utilization was higher than the neighboring datum points (sensitivity runs). It appears that although the CPU utilization increased, during a few sensitivity runs, there was a consistent decline in the gain factor (degradation of the throughput or turnaround time). From the results obtained, it appears that a change in the workload will allow an increase in the CPU utilization but poorer turnaround results will also arise for the AMIS users.

2) The graph on page 64 supports the idea that the gain factor lowers in conjunction with an increase in CPU utilization. The gain factor for the baseline run was at a maximum, decreasing substantially as the interactive workload increased.

3) Never once were there 35 or more interactive terminals being used concurrently. From the results obtained and shown on page 65 a decrease in the number of interactive ports causes a 2% increase in CPU utilization, while at the same time causing the gain factor to fall by .3935.

4) If real time updates were to be performed interactively by AMIS users, the simulation results predict that throughput performance would probably decline. This premise is drawn from the data on page 64 where an increase in the interactive workload reflects a steady decline in the gain factor.

BIBLIOGRAPHY

1. Ackoff, R.L. and M. W. Sasieni. Fundamentals of Operational Research. John Wiley and Sons, Inc. New York (1968)

2. Agrawala, A.K. and J. M. Mohr. "Some Results on the Clustering Approach to Workload Modelling." Computer Performance Evaluation, 13th Meeting: NBS Spec Publ. 500-18, 23-29 (1977)

3. Anderberg, M.R. Cluster Analysis. Academic Press, New York and London (1973)

4. Auerbach Quide to Data Base Management. Auerbach Publishers, Philadelphia (1975)

5. Bell, T.E. "Objectives and Problems in Simulating Computers." AFIPS Proc. Fall Joint Computer Conference: 287-297 (1972)

6. Bell, T.E., Boehm, B.W. and R. A. Watson. "Framework and Initial Phases for Computer Performance Improvement." AFIPS Fall Joint Computer Conference: 1141-1154 (1972)

7. Blitt, W.J. and R. Heidenreich "The Advanced Logistics System Cyber 73: A Simulation Model." Air Force Insitute of Technology (December 1975) DDC# AD-A019 841

8. Buckley, J.E. "Remote Batch Vs Interactive Processing." Computer Design: 14, 10 (September 1975)

9. Buzen, J.P. "Fundamental Operational Laws of Computer System Performance." Acta Informatia: 7, 167-182 (1976)

10 Cloot, P.L. "What is the Use of Operating Systems?" Computer Journal: 7, No 2, 249-254 (July 1964)

11 Colella, A.M., Sullivan, M.J. and D. J. Carlino. System Simulation. Lexington Books, D.C. Health and Company (1974)

12 DeCegama, A. "A Methodology for Computer Model Building." AFIPS Proc. Fall Joint Computer Conference: 299-310 (1972)

13 Doherty, W.J. "Scheduling TSS/360 for Responsiveness." AFIPS Proc.Fall Joint Computer Conference: 97-111 (1970)

14 Donovan, J.J. and S.E. Madnick. Operating Systems. Mc Graw-Hill Book Company, (1974)

15 Douglas, W. and J. E. Shemer. "Performance Modeling and Empirical Measurements in a System Designed for Batch and Time-Sharing Users." AFIPS Proc. Fall Joint Computer Conference: 17-26 (1969)

16 Drummond, M.E. "A Perspective on System Performance Evaluation." IBM System Journal: No 4, 252-263 (1969)

17. Forgy, E.W. "Cluster Analysis of Multivariate Data: Efficiency Versus Interpretability of Classifications." Biometric Soc. Meetings Riverside, Calif (Abstract in Biometrics: 21, No 3, 768) (1968)

18. Forbes, K. and A. W. Goldsworthy. "A Prescheduling Algorithm - Scheduling a Suitable Mix Prior to Processing." Computer Journal: 20, 27-29 (February 1977)

19. Green, P.E., Frank, R.E. and P. J. Robinson. "Cluster Analysis in Test Market Selection." Management Science: 13, No 2, B-387-B-399 (April 1976)

20. Grower, J.C. "A Comparison of Some Methods of Cluster Analysis." Biometrics: 23, 623-638 (1967)

21. IBM, 1: System/370:Principles of Operation, Form GA22-7000.

22. IBM, 2: System/370: System Summary, Form GA22-7001-1.

23. Kay, I.M. "GPSS/Simscript - The Dominant Simulation Languages." Souther Simulation Service. Inc. Thomas M. Kisko, University of Florida

24. Kleinrock, L. and R. R. Muntz. "Processor Sharing Queueing Models of Mixed Scheduling Disciplines for Time Shared Systems." Journal of the Association for Computing Machinery: 19, 464-482 (July 1972)

25. Lance, G.N. and W. T. Williams. "A General Theory of Classification Sorting Strategies 1. Hierarchical Systems." Computer Journal 9, No 4 373-380 (February 1967)

26. Mac Dougall, M.H. "Computer System Simulation: An Introduction." Computing Surveys: Vol. 2. No 3 (September 1970)

27. Naylor, T.H. The Design of Computer Simulation Experiments. Duke University Press, Durham, N.C.(1969)

28. Shannon, R. System Simulation: The Art and Science. Prentice-Hall, Inc. Englewood Cliffs, New Jersey (1975)

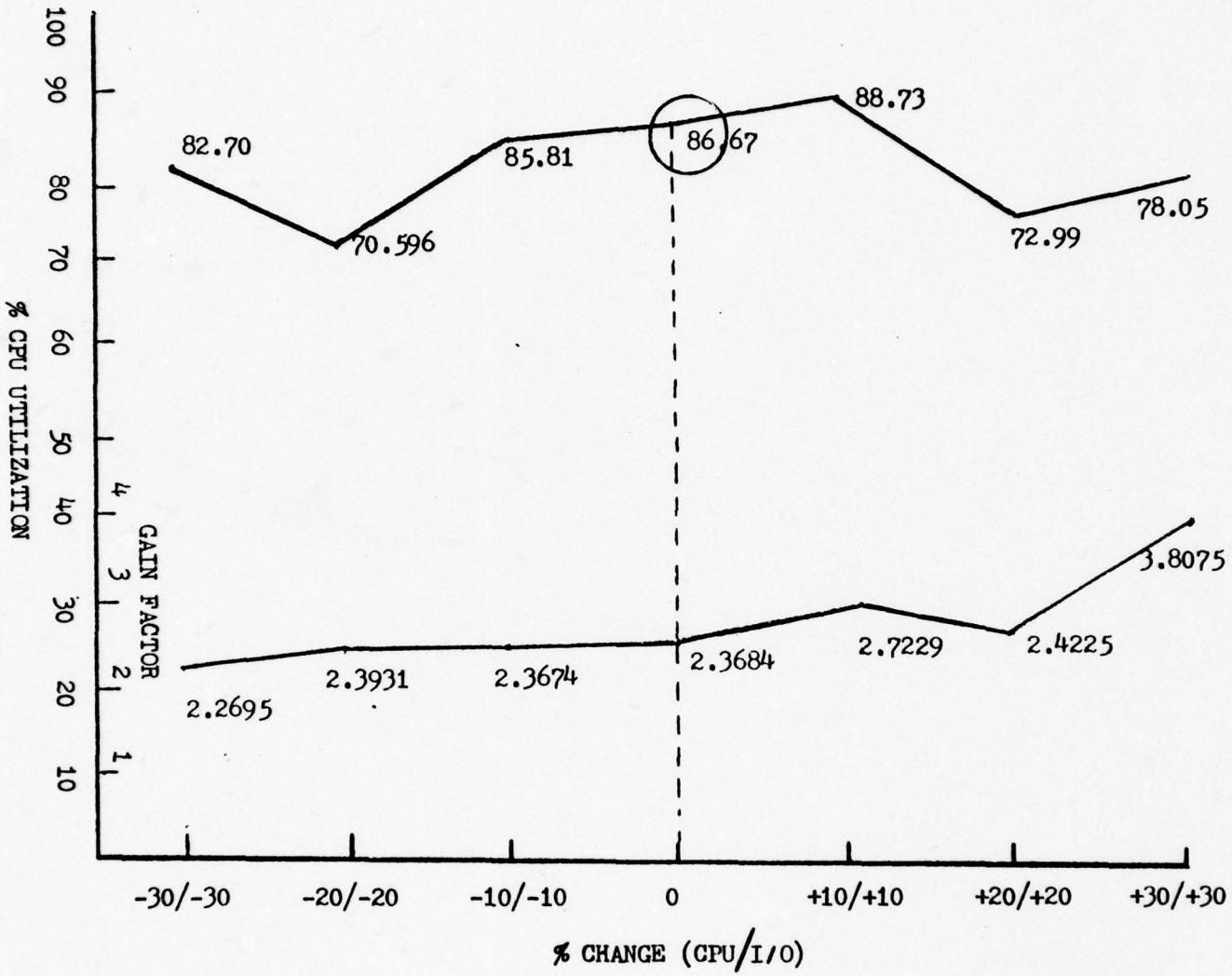
29. Simscript II.5 Reference Handbook. C.A.C.I. (1976)

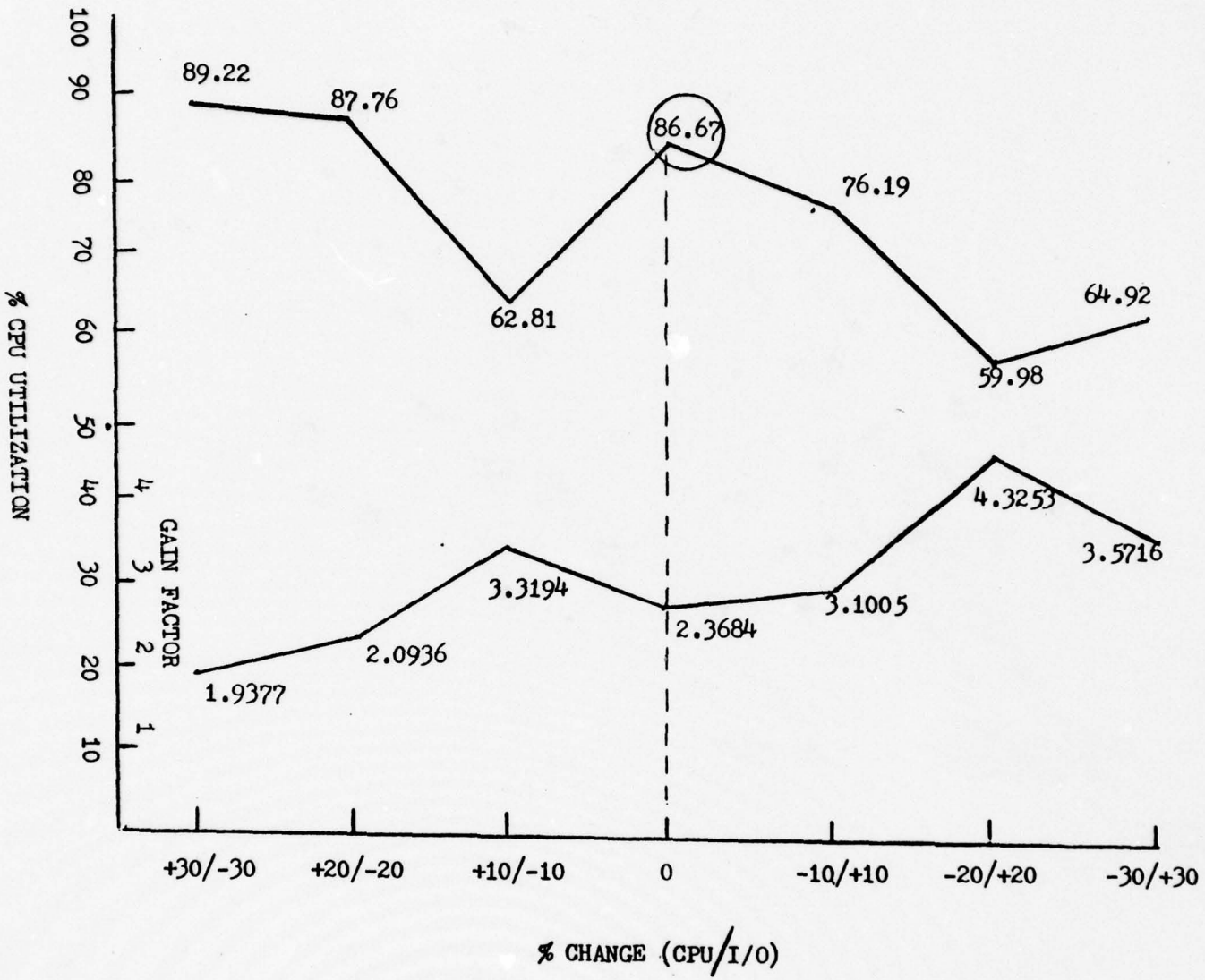
30. Svobodoba, L. Computer Performance Measurement and Evaluation Methods: Analysis and Applications. American Elsevier Publishing Company, Inc. (1976)

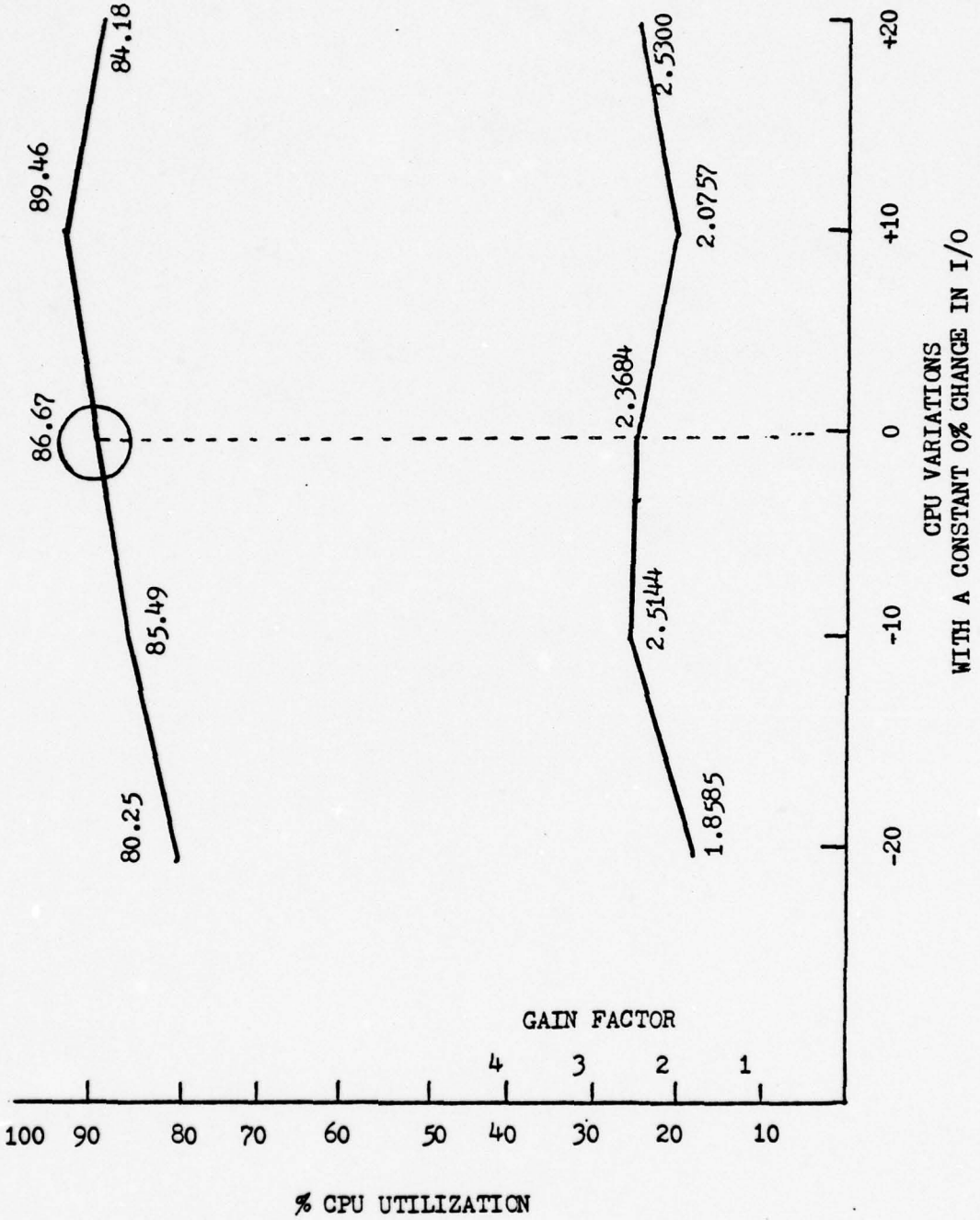
31. Wright, W.E. "An Axiomatic Specification of Euclidean Analysis." Computer Journal: 17, 355-364 (March 1973)

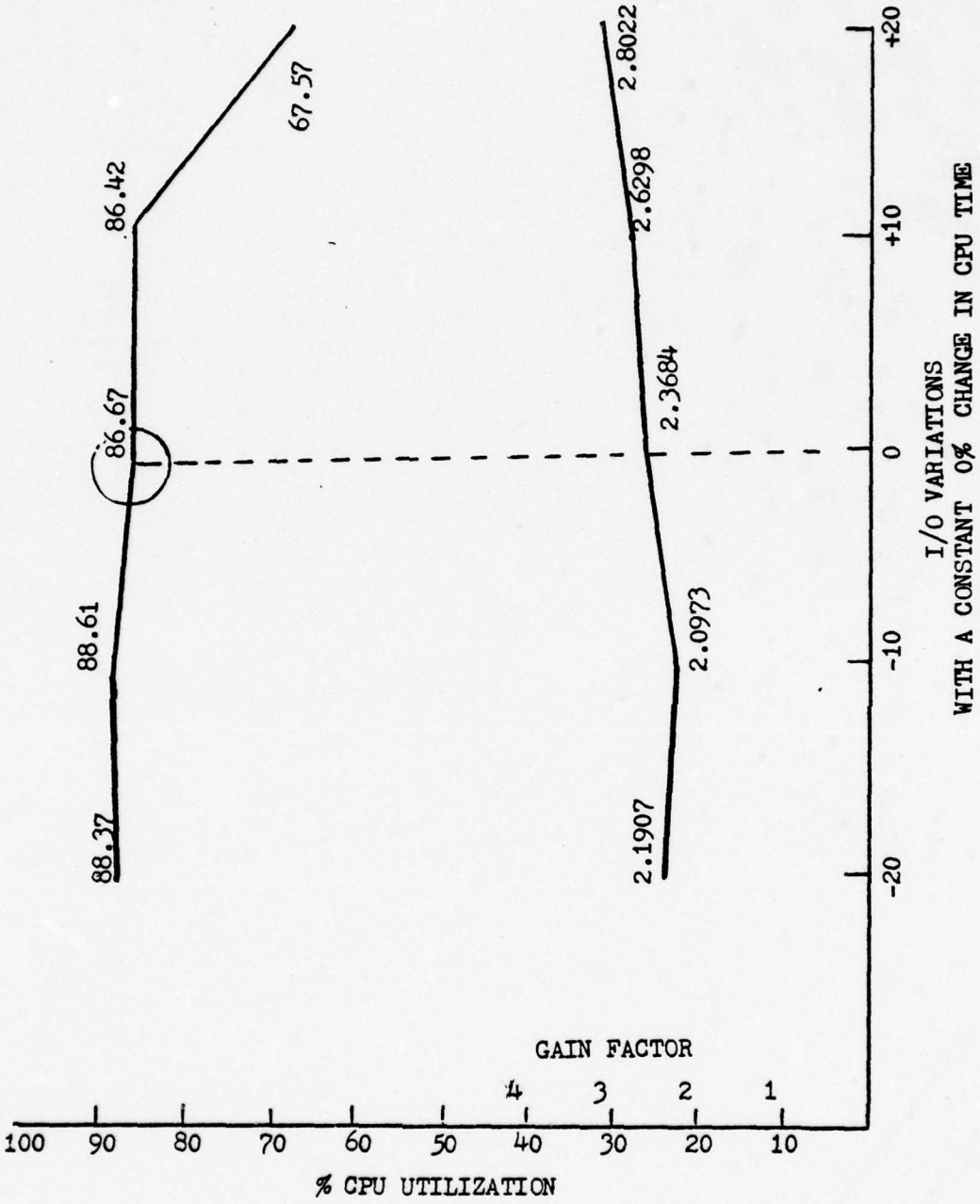
A P P E N D I X A

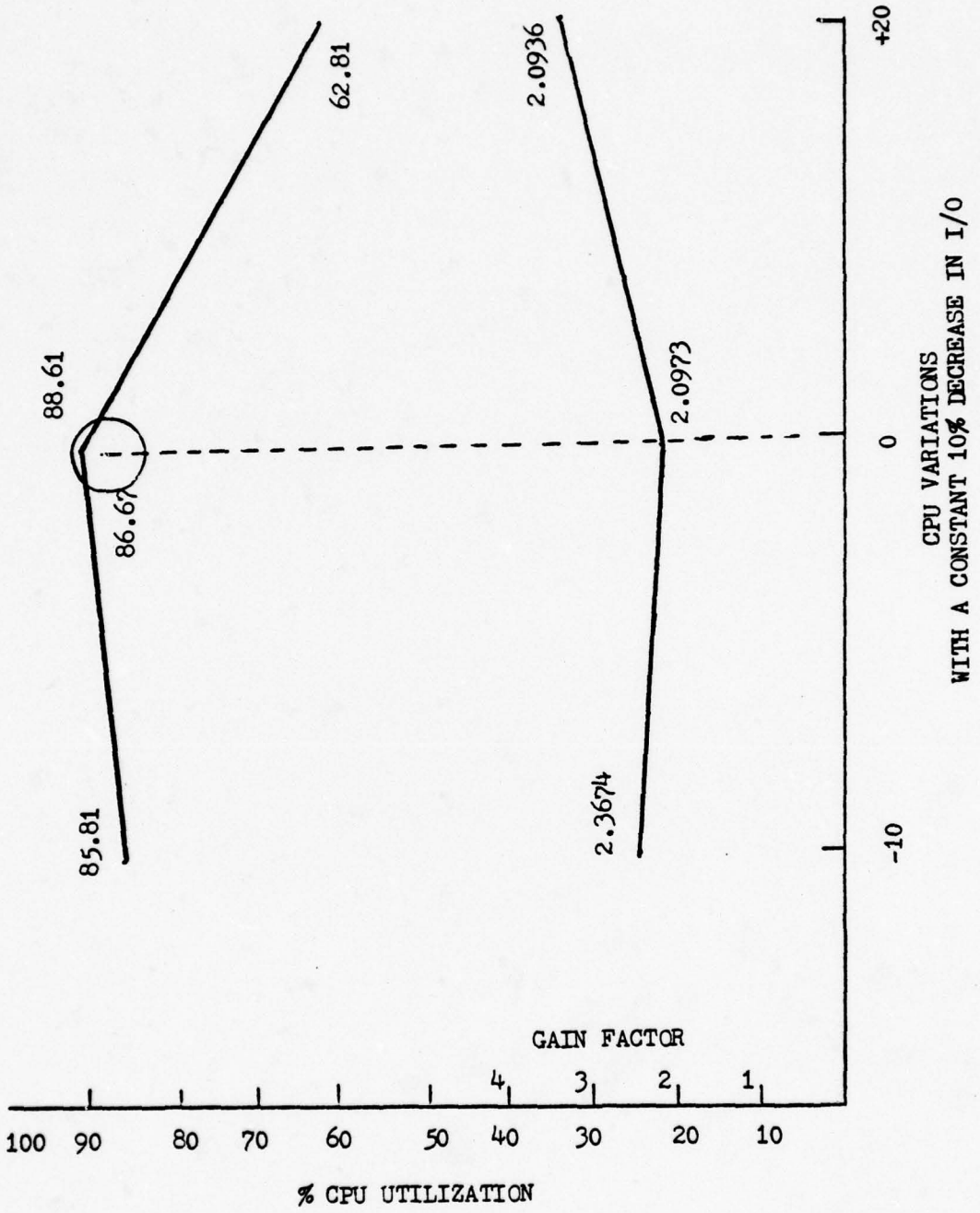
CPU Utilization vs Gain Factor

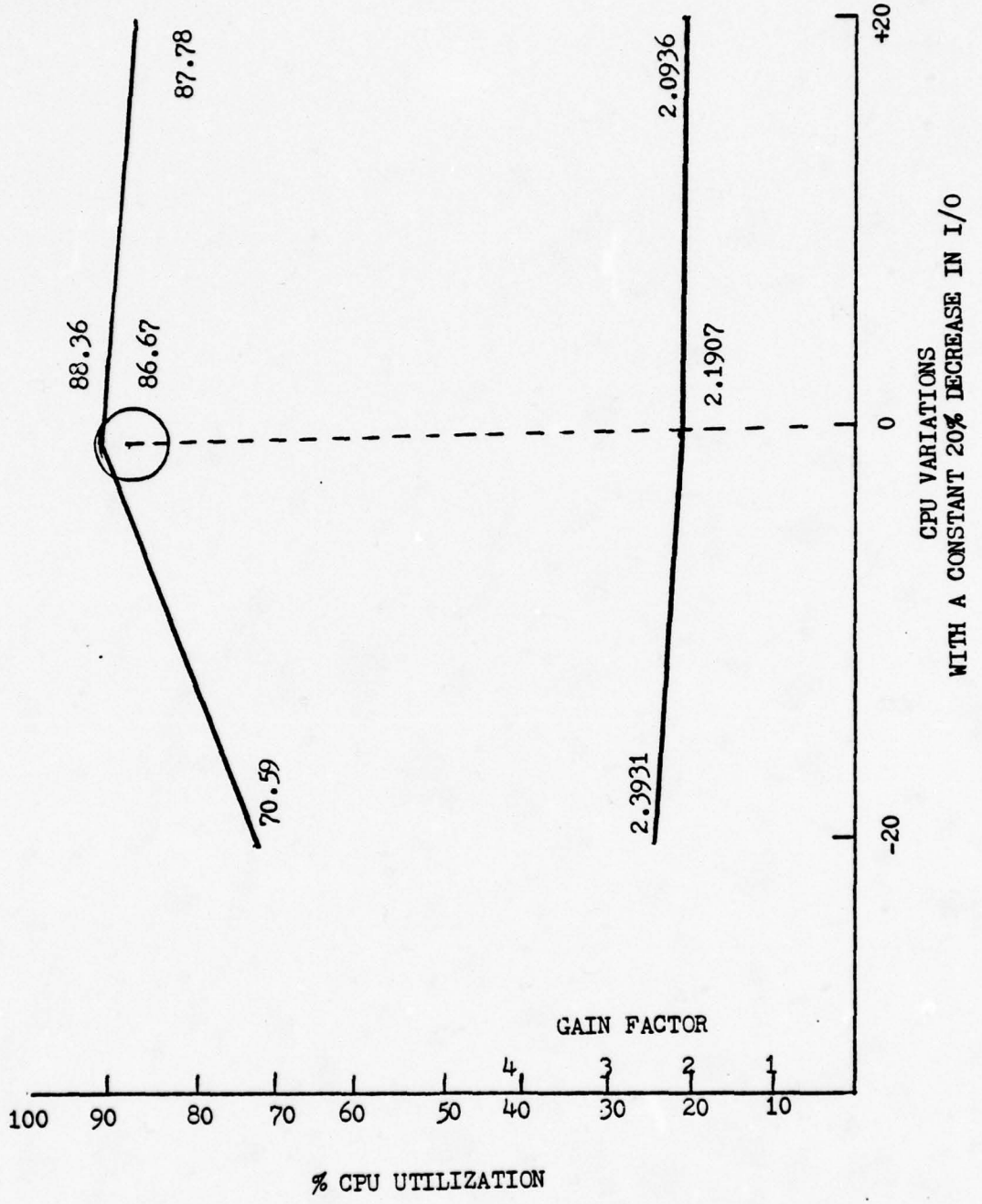

100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION
4
3
2
1
GAIN FACTOR
82.70
70.596
85.81
86.67
88.73
72.99
78.05
2.2695
2.3931
2.3674
2.3684
2.7229
2.4225
3.8075
-30/-30
-20/-20
-10/-10
0
+10/+10
+20/+20
+30/+30
% CHANGE (CPU/I/O)

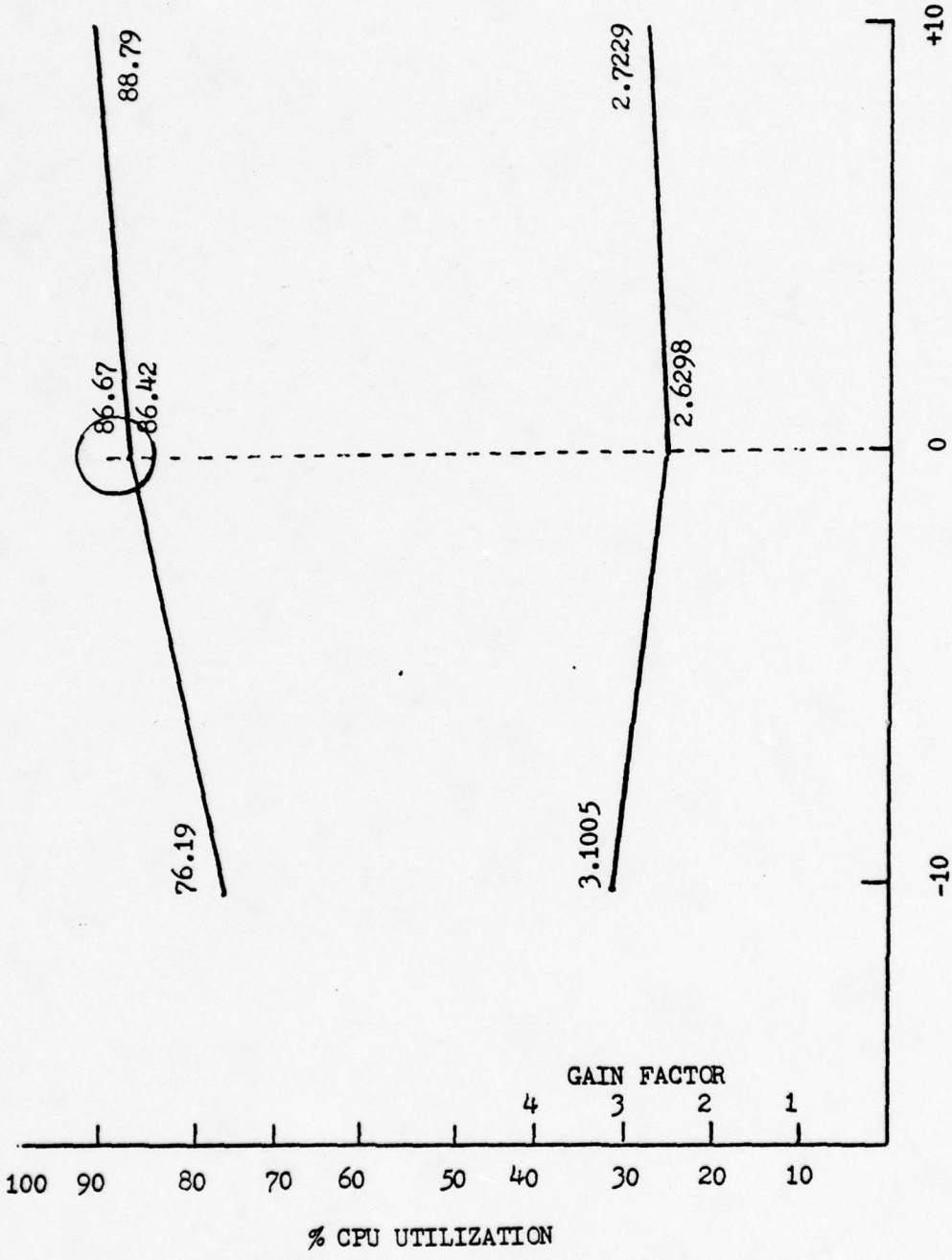

100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION
4
3
2
1
GAIN FACTOR
89.22
87.76
62.81
86.67
76.19
59.98
64.92
1.9377
2.0936
3.3194
2.3684
3.1005
4.3253
3.5716
+30/-30
+20/-20
+10/-10
0
-10/+10
-20/+20
-30/+30
% CHANGE (CPU/I/O)


89.46
84.18
86.67
85.49
80.25
2.5300
2.0757
2.3684
2.5144
1.8585
+20
+10
0
-10
-20
CPU VARIATIONS
WITH A CONSTANT 0% CHANGE IN I/O
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

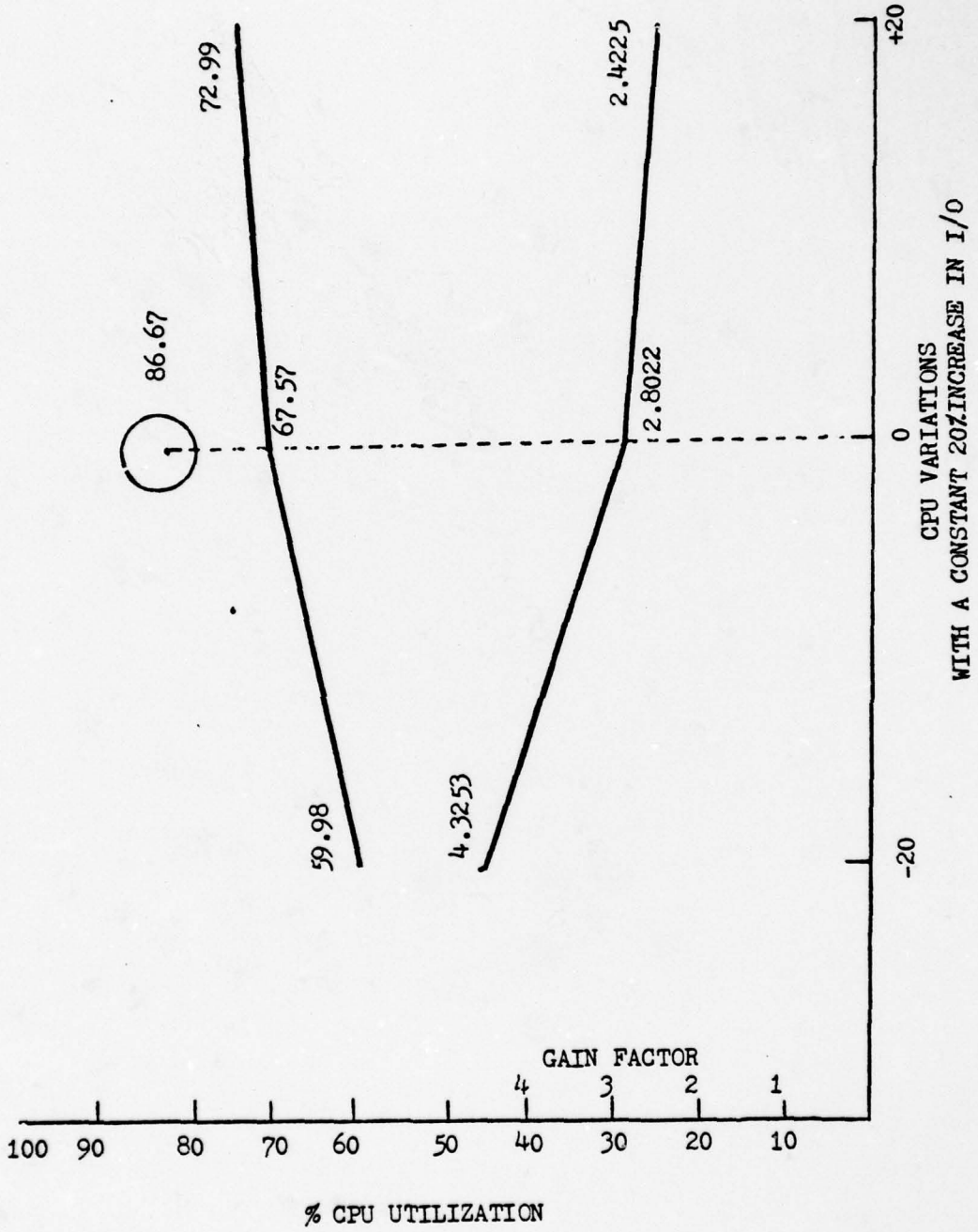

I/O VARIATIONS
WITH A CONSTANT 0% CHANGE IN CPU TIME
+20
+10
0
-10
-20
2.8022
2.6298
2.3684
2.0973
2.1907
67.57
86.42
86.67
88.61
88.37
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

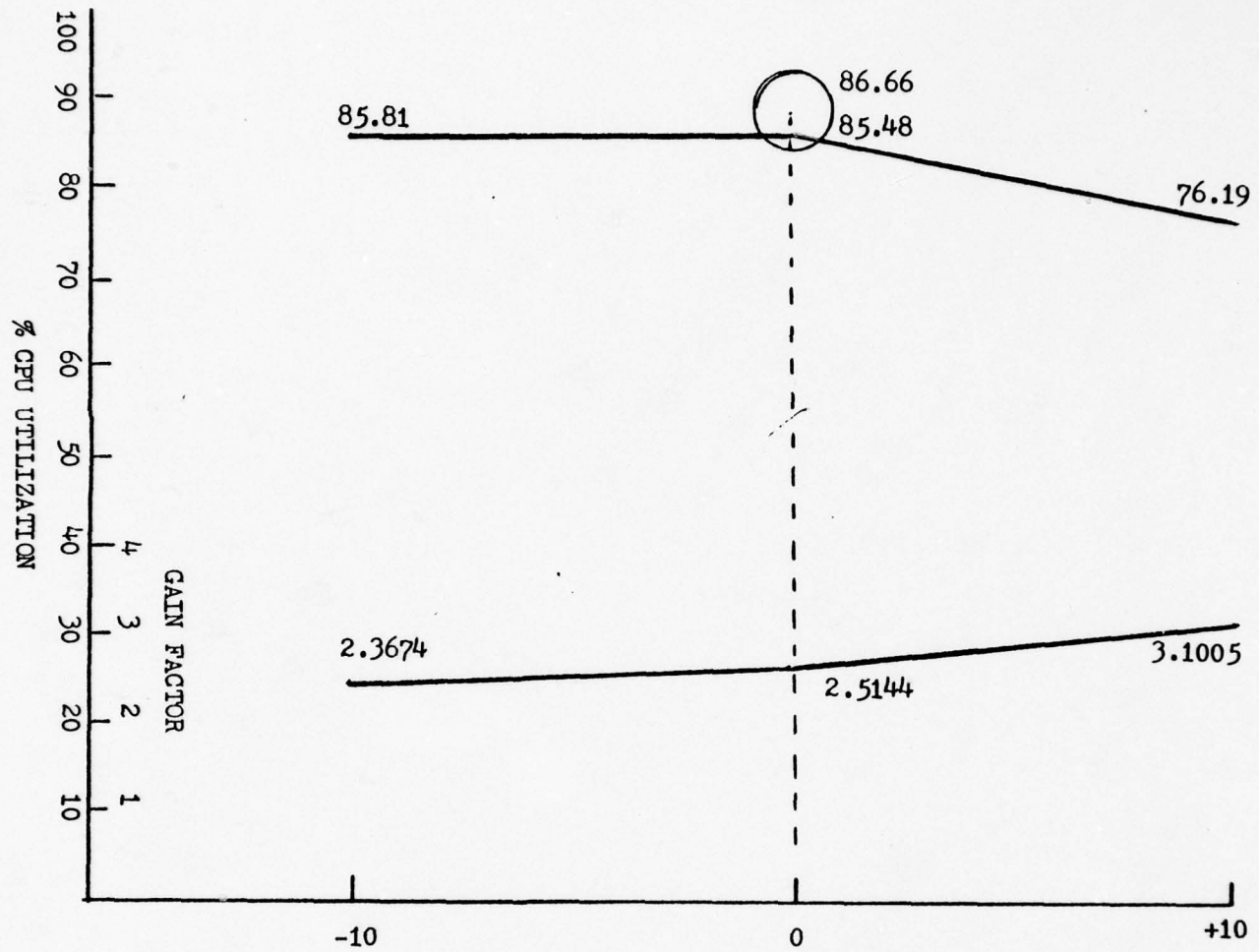

+20
0
-10
CPU VARIATIONS
WITH A CONSTANT 10% DECREASE IN I/O
62.81
2.0936
88.61
2.0973
86.67
85.81
2.3674
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

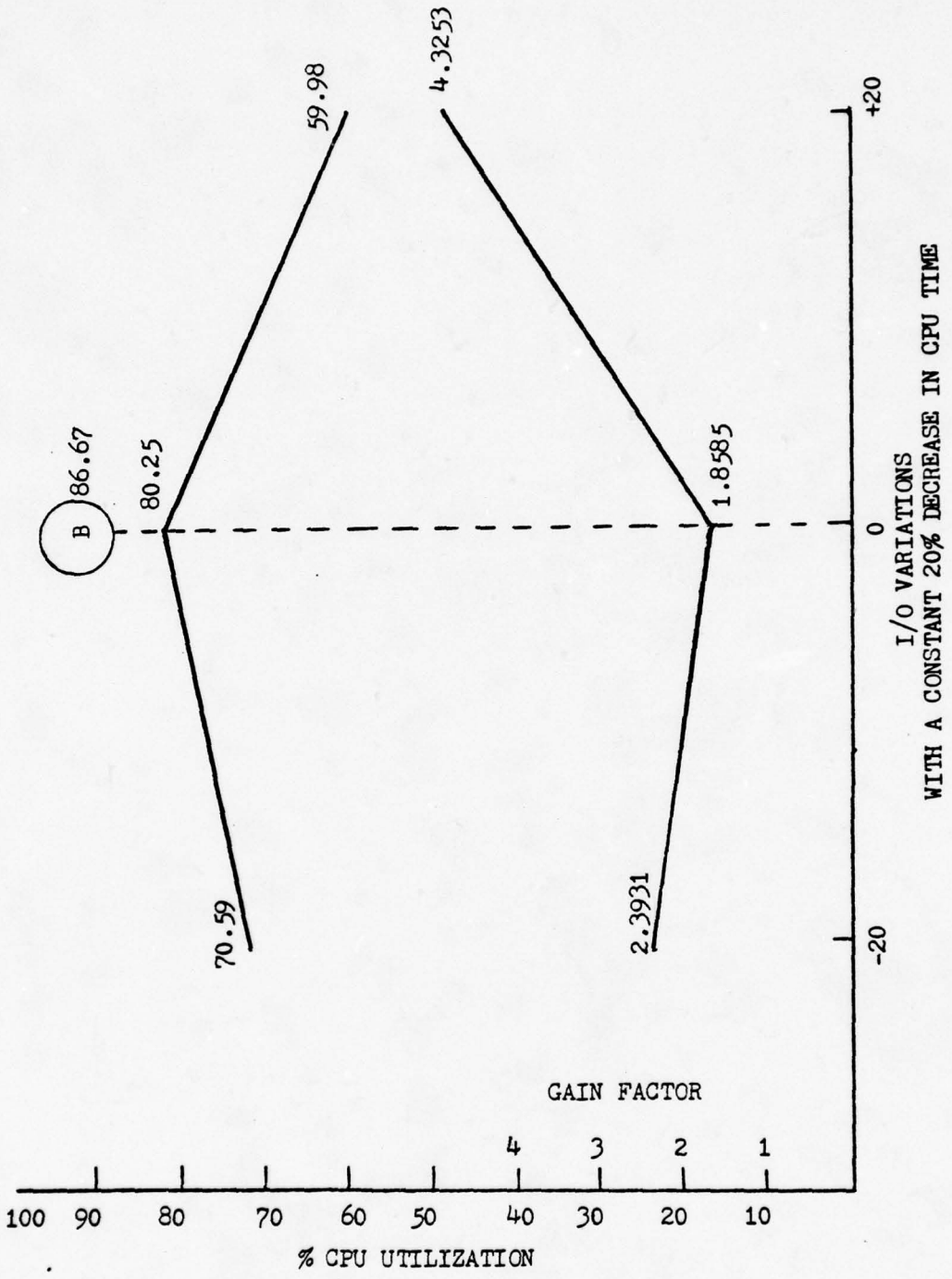

88.36
86.67
87.78
70.59
1
2.0936
2.1907
2.3931
+20
0
-20
CPU VARIATIONS
WITH A CONSTANT 20% DECREASE IN I/O
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

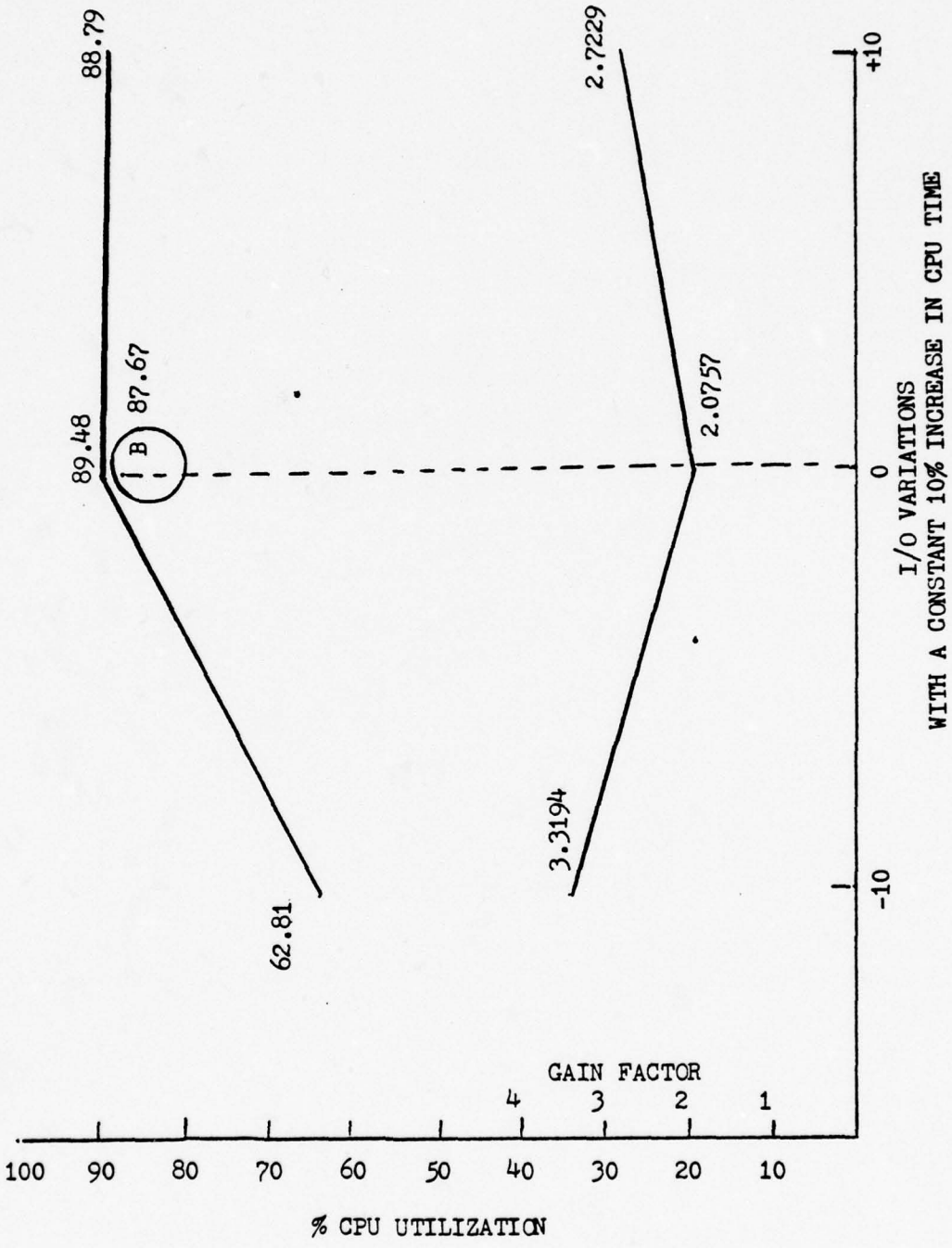


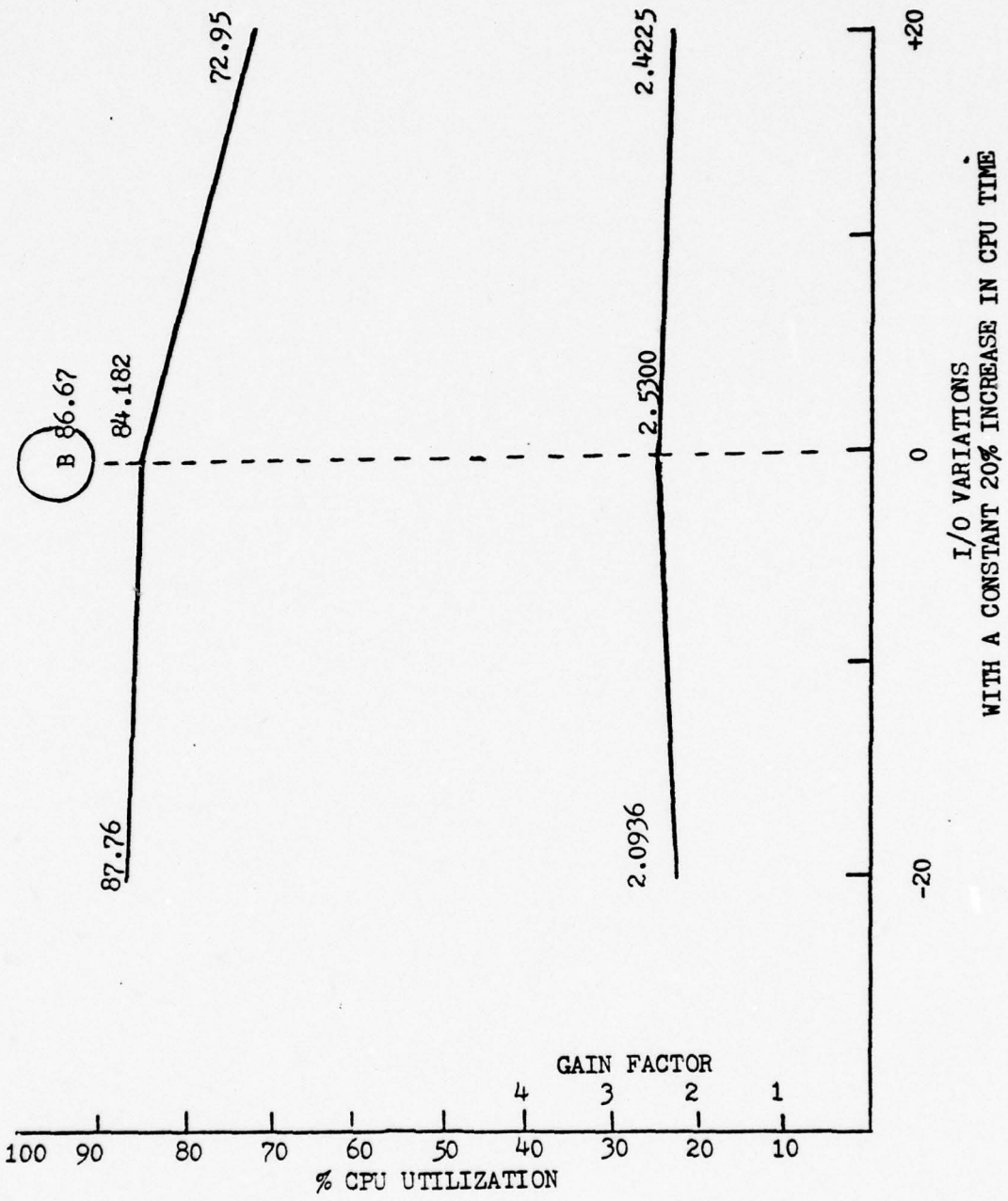
CPU VARIATIONS
WITH A CONSTANT 10% INCREASE IN I/O

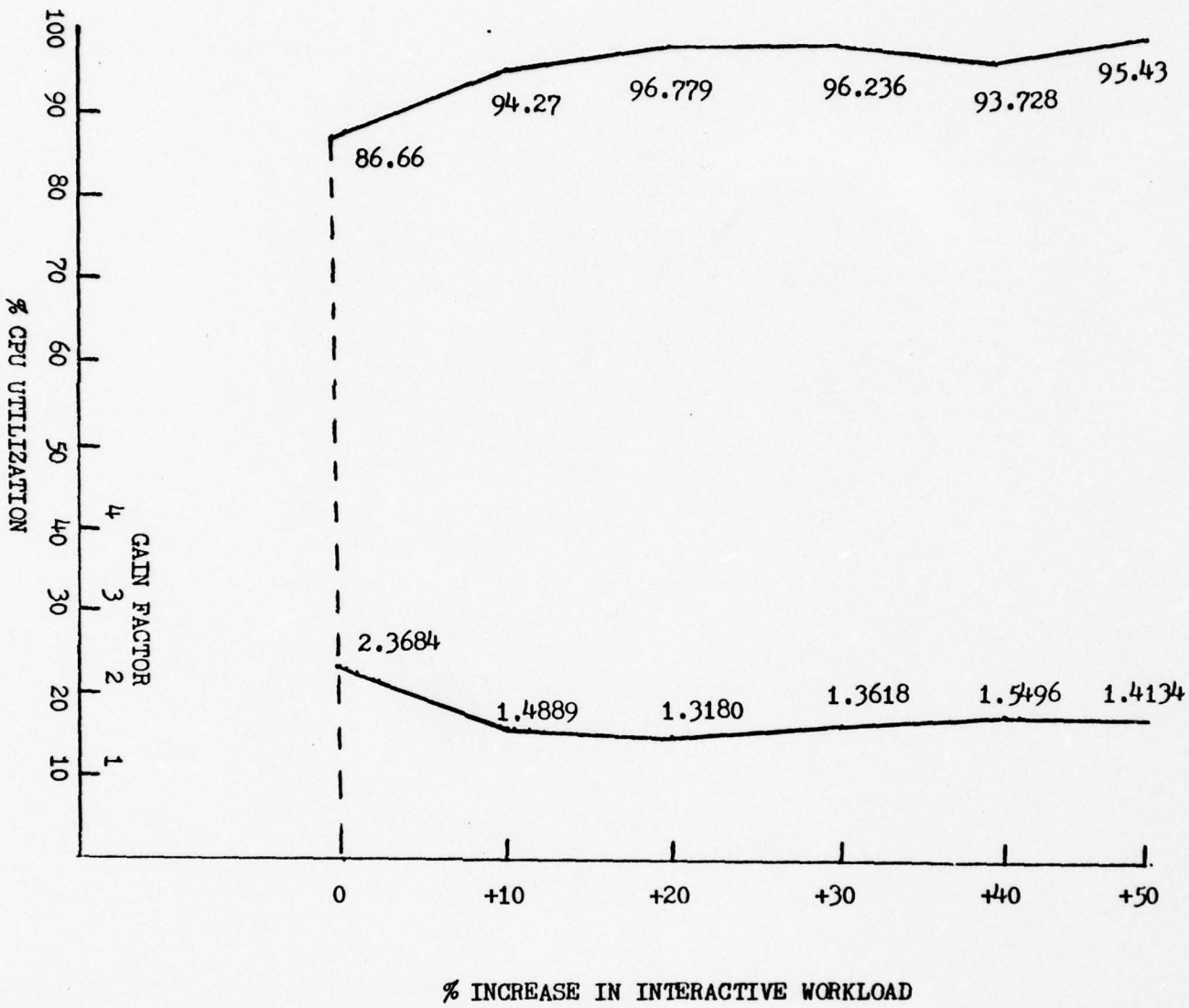

72.99
2.4225
86.67
67.57
2.8022
59.98
4.3253
+20
0
-20
CPU VARIATIONS
WITH A CONSTANT 20% INCREASE IN I/O
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

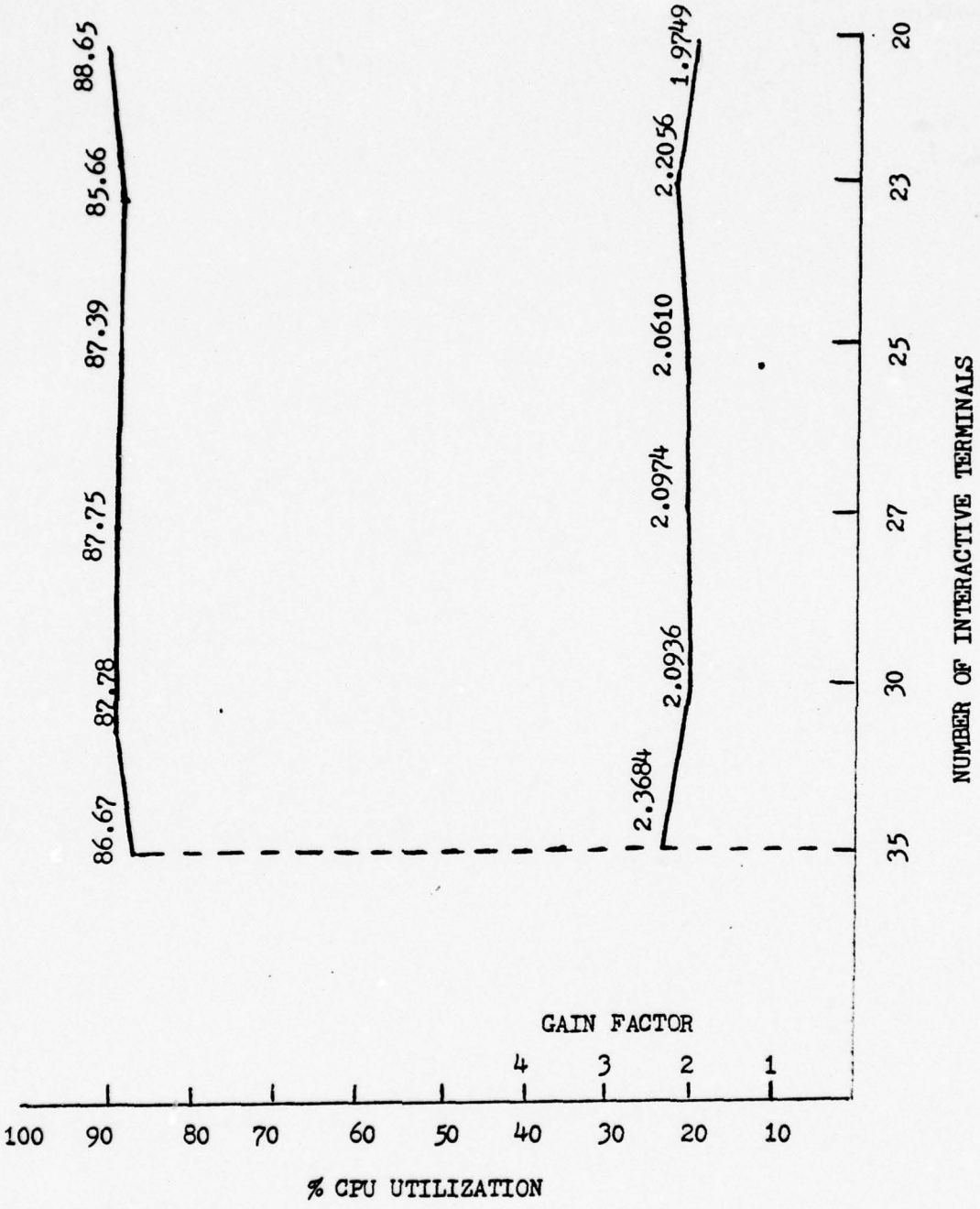

100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION
4
3
2
1
GAIN FACTOR
86.66
85.81
85.48
76.19
2.3674
3.1005
2.5144
-10
0
+10
I/O VARIATIONS
WITH A CONSTANT 10% DECREASE IN CPU TIME


B
86.67
80.25
59.98
70.59
4.3253
1.8585
2.3931
+20
0
-20
I/O VARIATIONS
WITH A CONSTANT 20% DECREASE IN CPU TIME
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION


88.79
89.48
87.67
B
62.81
2.7229
2.0757
3.3194
+10
0
-10
I/O VARIATIONS
WITH A CONSTANT 10% INCREASE IN CPU TIME
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION


72.95
2.4225
B 86.67
84.182
2.5300
87.76
2.0936
+20
0
-20
I/O VARIATIONS
WITH A CONSTANT 20% INCREASE IN CPU TIME
GAIN FACTOR
4 3 2 1
100 90 80 70 60 50 40 30 20 10
% CPU UTILIZATION


100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION
4
3
2
1
GAIN FACTOR
86.66
94.27
96.779
96.236
93.728
95.43
2.3684
1.4889
1.3180
1.3618
1.5496
1.4134
0
+10
+20
+30
+40
+50
% INCREASE IN INTERACTIVE WORKLOAD


88.65
85.66
87.39
87.75
82.28
86.67
1.9749
2.2056
2.0610
2.0974
2.0936
2.3684
20
23
25
27
30
35
NUMBER OF INTERACTIVE TERMINALS
GAIN FACTOR
4
3
2
1
100
90
80
70
60
50
40
30
20
10
% CPU UTILIZATION

A P P E N D I X B

Simscript Simulation Program

This computer program in Appendix B took nearly 60 seconds to compile and each run took about 185 seconds for execution time. The ratio of simulation time to execution time (time to execute each run) is approximately 80 to 1. (It took 185 seconds to execute the simulation program for a simulated period of 4 hours). The model itself is comprised of 23 separate events (similar to subroutines) beginning with the preamble.

The different job attributes, events and parameters are established in the preamble while the actual values are established in the main routine. The variables of interest in each event are given a short discription within the event that they occur. Generally these discriptions are placed between two lines of "stars" which are physically placed near the beginning of each event or routine.

```
PREAMBLE                                                                    000100
**************************************************************************  000110
**JT IS THE JOB TYPE VARIABLE                                            *  000120
**                                                                       *  000130
**CPU.SF IS THE CPU STATUS FLAG                                          *  000140
**       0 = CPU IDLE     1 = CPU BUSY                                   *  000150
**                                                                       *  000160
**NBJ IS THE NUMBER OF BATCH JOBS IN THE SYSTEM                          *  000170
**                                                                       *  000180
**NIJ IS THE NUMBER OF INTERACTIVE JOBS IN THE SYSTEM                    *  000190
**                                                                       *  000200
**NJNPLO IS THE NUMBER OF INTERACTIVE JOBS NOT PERMITTED TO LOGON        *  000210
**                                                                       *  000220
**SF.DISK IS THE DISK STATUS FLAG                                        *  000230
**     0 = DISK IDLE     1 = BUSY                                        *  000240
**                                                                       *  000250
**SECOND IS EQUAL TO 1.0/60.0                                            *  000260
**                                                                       *  000270
**MAIN.MEMORY IS THE MAIN MEMORY CORE ARRAY                              *  000280
**                                                                       *  000290
**JIS IS THE NUMBER OF JOB INITIATORS AVAILABLE FOR BATCH JOBS           *  000300
**JISI  IS THE NUMBER OF JOB INITIATORS AVAILABLE FOR INTERACTIVE JOBS       000310
**                                                                       *  000320
**JOB.SIZE INDICATES AMONT OF MEMORY NEEDED                              *  000330
**                                                                       *  000340
**CP.T.QUANTUM IS THE TIME QUANTUM FOR BATCH OR INTERACTIVE JOB          *  000350
**NEEDED.RESOURCES.F IS A NUMERICAL INDICATION OF RESOURCES NEEDED       *  000360
**P.OF.RESOURCES IS THE NUMERICAL REPRESENTATION OF RESOURCE WAIT        *  000370
**          TIME USING P&V SEMAPHORES                                    *  000380
**GAIN = CP.TIME(AMIS) + IO.TIME(AMIS)                                   *  000390
**************************************************************************  000400
NORMALLY MODE IS INTEGER                                                    000410
DEFINE P.SF AS INTEGER VARIABLE                                             000420
DEFINE N AS INTEGER VARIABLE                                                000430
DEFINE K AS INTEGER VARIABLE                                                000440
DEFINE FLAG AS INTEGER VARIABLE                                             000450
```

```
DEFINE STATUS.PRIORITY AS INTEGER VARIABLE                          000460
DEFINE DISK.MOUNT AS INTEGER VARIABLE                               000470
DEFINE TAPE.MOUNT AS INTEGER VARIABLE                               000480
DEFINE JT AS INTEGER VARIABLE                                       000490
DEFINE CPU.SF AS INTEGER VARIABLE                                   000500
DEFINE NBJ AS INTEGER VARIABLE                                      000510
DEFINE JISI AS INTEGER VARIABLE                                     000520
DEFINE USER.RESPONSE.TIME AS INTEGER VARIABLE                       000530
DEFINE NSUI AS INTEGER VARIABLE                                     000540
DEFINE NSUB AS INTEGER VARIABLE                                     000550
DEFINE NIJ AS INTEGER VARIABLE                                      000560
DEFINE NJNPLO AS INTEGER VARIABLE                                   000570
DEFINE SF.DISK AS INTEGER VARIABLE                                  000580
DEFINE JIS AS INTEGER VARIABLE                                      000590
DEFINE JOB.SIZE AS INTEGER VARIABLE                                 000600
DEFINE WARM.UP.FLAG AS INTEGER VARIABLE                             000610
DEFINE FF AS INTEGER VARIABLE                                       000620
DEFINE CHANNEL AS INTEGER VARIABLE                                  000630
DEFINE LOAD.F AS INTEGER VARIABLE                                   000640
DEFINE JN AS INTEGER VARIABLE                                       000650
DEFINE RESOURCES.F AS INTEGER VARIABLE                              000660
DEFINE I.MEMORY.REQUEST AS INTEGER VARIABLE                         000670
DEFINE B.MEMORY.REQUEST AS INTEGER VARIABLE                         000680
DEFINE PFLAG AS INTEGER VARIABLE                                    000690
DEFINE PJOFLAG AS INTEGER VARIABLE                                  000700
DEFINE POUTFLAG AS INTEGER VARIABLE                                 000710
DEFINE PCMFLAG AS INTEGER VARIABLE                                  000720
DEFINE N.JOBS.PROCESSED AS INTEGER VARIABLE                         000730
DEFINE B.COUNTER AS INTEGER VARIABLE                                000740
DEFINE I.COUNTER AS INTEGER VARIABLE                                000750
DEFINE INUMBER AS INTEGER VARIABLE                                  000760
DEFINE BNUMBER AS INTEGER VARIABLE                                  000770
```

```
DEFINE TOT.JOB.IO.TIME AS REAL VARIABLE                          000840
DEFINE TOTAL.JOB.CPU.TIME AS REAL VARIABLE                       000850
DEFINE BATCH.IO AS REAL VARIABLE                                 000860
DEFINE TCPUT AS REAL VARIABLE                                    000870
DEFINE CU AS REAL VARIABLE                                       000880
DEFINE INTERACTIVE.IO AS REAL VARIABLE                           000890
DEFINE GAIN AS REAL VARIABLE                                     000900
DEFINE GAIN.FACTOR AS REAL VARIABLE                              000910
DEFINE IO.OVERHEAD.T AS REAL VARIABLE                            000920
DEFINE TS AS REAL VARIABLE                                       000930
DEFINE ETMF AS REAL VARIABLE                                     000940
DEFINE I.ETMF AS REAL VARIABLE                                   000950
DEFINE I.EX.DELAY.FACTOR AS REAL VARIABLE                        000960
DEFINE EX.DELAY.FACTOR AS REAL VARIABLE                          000970
DEFINE I.CAPABILITY AS REAL VARIABLE                             000980
DEFINE T AS REAL VARIABLE                                        000990
DEFINE B.RT AS REAL VARIABLE                                     001000
DEFINE I.RT AS REAL VARIABLE                                     001010
DEFINE I.TURNAROUND.T AS REAL VARIABLE                           001020
DEFINE B.TURNAROUND.T AS REAL VARIABLE                           001030
DEFINE W.UP.TIME AS REAL VARIABLE                                001040
DEFINE I.CPU.TIME AS REAL VARIABLE                               001050
DEFINE B.CPU.TIME AS REAL VARIABLE                               001060
DEFINE B.CAPABILITY AS REAL VARIABLE                             001070
DEFINE CURRENT.UTILIZATION AS REAL VARIABLE                      001080
DEFINE T.CPU.TIME AS REAL VARIABLE                               001090
DEFINE ST AS REAL VARIABLE                                       001100
DEFINE P AS REAL VARIABLE                                        001110
DEFINE SEC.DAYS AS REAL VARIABLE                                 001120
DEFINE MIL.SEC.DAYS AS REAL VARIABLE                             001130
DEFINE SECOND AS REAL VARIABLE                                   001140
DEFINE MS AS REAL VARIABLE                                       001150
DEFINE IODTT AS REAL VARIABLE                                    001160
DEFINE CP.T.QUANTUM AS REAL VARIABLE                             001170
DEFINE MAIN.MEMORY AS INTEGER,1-DIMENSIONAL ARRAY                001180
```

```
THE SYSTEM OWNS                                                          001190
      A TEMP.QUEUE,                                                      001200
      A LOAD.QUEUE,                                                      001210
      A WAIT.QUEUE,                                                      001220
      A THINK.STATE,                                                     001230
      A XX.BUFFER.QUEUE,                                                 001240
      A IO.QUEUE,                                                        001250
      A JOB.QUEUE,                                                       001260
      A CM.QUEUE,                                                        001270
      A CPU..REQUEST.QUEUE,                                              001280
      A AN.OUTPUT.QUEUE,                                                 001290
      A ACM.RELEASE.QUEUE,                                               001300
      A HH.OUTPUT.QUEUE,                                                 001310
      AND A ACPU                                                         001320
                                                                         001330
                                                                         001340
**************************************************************************001350
**                                                                      *001360
**XX.TOTAL.CPU.TIME IS THE TOTAL TIME REQUESTED BY A SINGLE JOB         *001370
**                                                                      *001380
**ACC.CPU.TIME IS THE TIME THE JOB ACCESSED THE CPU                     *001390
**                                                                      *001400
**JOB.IO.REQUESTS IS THE TOTAL NUMBER OF I/O OPERATIONS REQUESTED       *001410
**           BY A SINGLE JOB                                           *001420
**                                                                      *001430
**TLF IS THE TIME LEFT FLAG                                            *001440
**     =0 MEANS THERE IS TIME LEFT OVER FOR CURRENT CPU TASK            001450
**     =1 MEANS THE CPU TASK HAS BEEN FINISHED                          *001460
**                                                                      *001470
**XX.CPU.SERVICE TIME IS THE CPU TIME REQUIRED TO PROCESS UNTIL         *001480
**          I/O TASK                                                    *001490
**                                                                      *001500
**IO.SERVICE TIME IS THE I/O TIME REQUIRED TO PROCESS A SINGLE IO TASK  001510
**                                                                      *001520
**T.LEFT.IN.CPU IS THE AMOUNT OF TIME REMAINING UNTIL NEXT I/O          *001530
**                                                                      *001540
**MEMORY.REQUEST IS THE AMONT OF MEMORY REQURESTED BY A SINGLE JOB      *001550
**************************************************************************001560
```

```
                                                                      001570
                                                                      001580
TEMPORARY ENTITIES                                                    001590
                                                                      001600
EVERY JOB HAS                                                         001610
      A TURNAROUND,                                                   001620
      A CP.TIME,                                                      001630
      A IO.TIME,                                                      001640
      A RE.SOURCES,                                                   001650
      A MEMORY.REQUEST,                                               001660
      A T.E.CPU,                                                      001670
      A START.MEMORY,                                                 001680
      A END.MEMORY,                                                   001690
      A JOBNUM,                                                       001700
      A JOB.TYPE,                                                     001710
      A TIME.E.CPJ,                                                   001720
      A ARRIVAL.TIME,                                                 001730
      A JOB.EXIT.TIME,                                                001740
      A PRIORITY,                                                     001750
      A T.LEFT.IN.CPU,                                                001760
      A NEEDED.RESOURCES.F,                                           001770
      A P.OF.RESOURCES,                                               001780
      A XXTIME.E.CPU.QUEUE,                                           001790
      A NO.CAR.RETURNS,                                               001800
      A F.LOADER,                                                     001810
      A LOADING.T,                                                    001820
      A PAGE.LENGTH,                                                  001830
      A XX..CPU.SERVICE.TIME,                                         001840
      A XX.CPU.SERVICE.TIME,                                          001850
      A JOB.IO.REQUESTS,                                              001860
      A JOB..IO.REQUESTS,                                             001870
      A JIO,                                                          001880
      A BLOCKED.TIME,                                                 001890
      A TLF,                                                          001900
      A IO.SERVICE.TIME,                                              001910
                                                                      001920
```

```
AND BELONGS TO                                                          001930
      A LOAD.QUEUE,                                                     001940
      A THINK.STATE,                                                    001950
      A WAIT.QUEUE,                                                     001960
      A JOB.QUEUE,                                                      001970
      A CM.QUEUE,                                                       001980
      A IO.QUEUE,                                                       001990
      A CPU..REQUEST.QUEUE,                                             002000
      A TEMP.QUEUE,                                                     002010
      A ACM.RELEASE.QUEUE,                                              002020
      A AN.OUTPUT.QUEUE,                                                002030
      A XX.BUFFER.QUEUE,                                                002040
      A HH.OUTPUT.QUEUE,                                                002050
      AND A ACPU                                                        002060
                                                                        002070
NORMALLY MODE IS REAL                                                   002080
DEFINE TURNAROUND AS REAL VARIABLE                                      002090
DEFINE T.E.CPU AS REAL VARIABLE                                         002100
DEFINE JOB.EXIT.TIME AS REAL VARIABLE                                   002110
DEFINE IO.TIME AS REAL VARIABLE                                         002120
DEFINE CP.TIME AS REAL VARIABLE                                         002130
DEFINE ARRIVAL.TIME    AS REAL VARIABLE                                 002140
DEFINE BLOCKED.TIME AS REAL VARIABLE                                    002150
DEFINE T.LEFT.IN.CPU AS REAL VARIABLE                                   002160
DEFINE XX.CPU.SERVICE.TIME AS REAL VARIABLE                             002170
DEFINE IO.SERVICE.TIME AS REAL VARIABLE                                 002180
DEFINE XX..CPU.SERVICE.TIME AS REAL VARIABLE                            002190
DEFINE XXTIME.E.CPU.QUEUE AS REAL VARIABLE                              002200
DEFINE LOADING.T AS REAL VARIABLE                                       002210
DEFINE CPU.R AS REAL VARIABLE                                           002220
                                                                        002230
NORMALLY MODE IS INTEGER                                                002240
                                                                        002250
DEFINE HH.OUTPUT.QUEUE AS FIFO SET                                      002260
DEFINE LOAD.QUEUE AS FIFO SET                                           002270
DEFINE XX.BUFFER.QUEUE AS FIFO SET                                      002280
```

```
DEFINE TEMP.QUEUE AS FIFO SET                                              002290
DEFINE THINK.STATE AS FIFO SET                                             002300
DEFINE CPU..REQUEST.QUEUE AS A FIFO SET RANKED BY                          002310
HIGH PRIORITY                                                              002320
DEFINE CM.QUEUE AS A FIFO SET RANKED BY HIGH PRIORITY                      002330
DEFINE JOB.QUEUE AS A FIFO SET RANKED BY HIGH PRIORITY                     002340
DEFINE WAIT.QUEUE AS A FIFO SET                                            002350
DEFINE IO.QUEUE AS FIFO SET                                                002360
DEFINE ACPU AS A FIFO SET                                                  002370
DEFINE ACM.RELEASE.QUEUE AS A FIFO SET                                     002380
DEFINE AN.OUTPUT.QUEUE AS A FIFO SET RANKED BY HIGH PRIORITY               002390
                                                                           002400
EVENT NOTICES INCLUDE                                                      002410
      DIAGNOSTIC,                                                          002420
      LOAD.DELAY,                                                          002430
      B.JOB.REQUEST,                                                       002440
      I.JOB.REQUEST,                                                       002450
      REQUEST.CM,                                                          002460
      XX.TRAFFIC.CONTROLLER,                                               002470
      RELEASE.CPU,                                                         002480
      CM.RELEASE,                                                          002490
      CPU.PROCESSING,                                                      002500
      AN.DISK.RELEASE,                                                     002510
      IS.BUFFER.EMPTY,                                                     002520
      RESPONSE,                                                            002530
      CPU.UTILIZATION,                                                     002540
      IO.DELAY,                                                            002550
      CPU.REQUEST,                                                         002560
      OUTPUT,                                                              002570
      XX.OUTPUT,                                                           002580
      P.INCREASE,                                                          002590
      PJO.INCREASE,                                                        002600
      P.OUTP.INCREASE,                                                     002610
      PCM.INCREASE,                                                        002620
      CPU.ANALYSIS,                                                        002630
      AND CEASE.SIM                                                        002640
```

```
EVERY RESPONSE HAS A RNAME                                              002650
EVERY IO.DELAY HAS A NAME                                               002660
TALLY IIO AS THE MEAN OF INTERACTIVE.IO                                 002670
TALLY BIO AS THE MEAN OF BATCH.IO                                       002680
TALLY ANSUI AS THE MEAN OF NSUI                                         002690
TALLY ENSUB AS THE MAX OF NSUB                                          002700
TALLY FNSUB AS THE MIN OF NSUB                                          002710
TALLY BNSUI AS THE MAX OF NSUI                                          002720
TALLY CNSUI AS THE MIN OF NSUI                                          002730
TALLY DNSUB AS THE MEAN OF NSUB                                         002740
TALLY AI.TURNAROUND.T AS MEAN OF I.TURNAROUND.T                         002750
TALLY BI.TURNAROUND.T AS VARIANCE OF I.TURNAROUND.T                     002760
TALLY CI.TURNAROUND.T AS STD.DEV OF I.TURNAROUND.T                      002770
TALLY AB.TURNAROUND.T AS MEAN OF B.TURNAROUND.T                         002780
TALLY BB.TURNAROUND.T AS VARIANCE OF B.TURNAROUND.T                     002790
TALLY CB.TURNAROUND.T AS STD.DEV OF B.TURNAROUND.T                      002800
TALLY AI.CPU.TIME AS MEAN OF I.CPU.TIME                                 002810
TALLY BI.CPU.TIME AS VARIANCE OF I.CPU.TIME                             002820
TALLY CI.CPU.TIME AS STD.DEV OF I.CPU.TIME                              002830
TALLY AB.CPU.TIME AS MEAN OF B.CPU.TIME                                 002840
TALLY BB.CPU.TIME AS VARIANCE OF B.CPU.TIME                             002850
TALLY CB.CPU.TIME AS STD.DEV OF B.CPU.TIME                              002860
TALLY AA AS MEAN OF I.CAPABILITY                                        002870
TALLY BB AS MEAN.SQUARE OF I.CAPABILITY                                 002880
TALLY CC AS VARIANCE OF I.CAPABILITY                                    002890
TALLY DD AS STD.DEV OF I.CAPABILITY                                     002900
TALLY EE AS MEAN OF B.CAPABILITY                                        002910
TALLY FFFF AS MEAN.SQUARE OF F.CAPABILITY                               002920
TALLY GG AS VARIANCE OF B.CAPABILITY                                    002930
TALLY HH AS STD.DEV OF B.CAPABILITY                                     002940
TALLY AAA AS THE MEAN OF I.BT                                           002950
TALLY BBB AS THE MEAN.SQUARE OF I.BT                                    002960
TALLY CCC AS THE VARIANCE OF I.BT                                       002970
TALLY DDD AS THE STD.DEV OF I.BT                                        002980
TALLY EEE AS THE MEAN OF B.BT                                           002990
TALLY FFF AS THE MEAN.SQUARE OF B.BT                                    003000
```

```
TALLY GGG AS THE VARIANCE OF B.BT                                    003010
TALLY HHH AS THE STD.DEV OF B.BT                                     003020
TALLY II AS THE MEAN OF I.MEMORY.REQUEST                             003030
TALLY JJ AS THE MEAN.SQUARE OF I.MEMORY.REQUEST                      003040
TALLY KK AS THE VARIANCE OF I.MEMORY.REQUEST                         003050
TALLY LL AS THE STD.DEV OF I.MEMORY.REQUEST                          003060
TALLY MM AS THE MEAN OF B.MEMORY.REQUEST                             003070
TALLY NN AS THE MEAN.SQUARE OF B.MEMORY.REQUEST                      003080
TALLY OO AS THE VARIANCE OF B.MEMORY.REQUEST                         003090
TALLY PP AS THE STD.DEV OF B.MEMORY.REQUEST                          003100
TALLY QQ AS THE MEAN OF USER.RESPONSE.TIME                           003110
TALLY RR AS THE MEAN.SQUARE OF USER.RESPONSE.TIME                    003120
TALLY SS AS THE VARIANCE OF USER.RESPONSE.TIME                       003130
TALLY TT AS THE STD.DEV OF USER.RESPONSE.TIME                        003140
TALLY UU AS THE MEAN OF ETMF                                         003150
TALLY VV AS THE VARIANCE OF ETMF                                     003160
TALLY WW AS THE STD.DEV OF ETMF                                      003170
TALLY XX AS THE MEAN.SQUARE OF ETMF                                  003180
TALLY UUU AS THE MEAN OF EX.DELAY.FACTOR                             003190
TALLY VVV AS THE VARIANCE OF EX.DELAY.FACTOR                         003200
TALLY WWW AS THE STD.DEV OF EX.DELAY.FACTOR                          003210
TALLY XXX AS THE MEAN.SQUARE OF EX.DELAY.FACTOR                      003220
TALLY I.E AS THE MEAN OF I.ETMF                                      003230
TALLY I.EA AS THE VARIANCE OF I.ETMF                                 003240
TALLY I.EB AS THE STD.DEV OF I.ETMF                                  003250
TALLY I.EC AS THE MEAN.SQUARE OF I.ETMF                              003260
TALLY I.EDEA AS THE MEAN OF I.EX.DELAY.FACTOR                        003270
TALLY I.EDEB AS THE VARIANCE OF I.EX.DELAY.FACTOR                    003280
TALLY I.EDEC AS THE STD.DEV OF I.EX.DELAY.FACTOR                     003290
TALLY I.EDED AS THE MEAN.SQUARE OF I.EX.DELAY.FACTOR                 003300
DEFINE TAPE.MOUNT TO MEAN TM                                         003310
DEFINE DISK.MOUNT TO MEAN DM                                         003320
END                                                                  003330
```

```
MAIN  ** INITIALIZE AND SCHEDULE FIRST JOBS                                    003410
PRINT 1 LINE THUS                                                              003420
ENTERED MAIN                                                                   003430
PRINT 1 LINE WITH TIME.V THUS                                                  003440
  SIMULATION TIME IS   ****.*****                                              003450
**************************************************************************     003460
      **NBJ IS THE NUMBER OF BATCH JOBS IN THE SYSTEM                    *      003470
      **NIJ IS THE NUMBER OF INTERACTIVE JOBS IN THE SYSTEM              *      003480
      **CPU.SF IS THE CPU STATUS FLAG     0=IDLE,1=BUSY                 *      003490
      **ST IS SCHEDULED JOB TIME                                        *      003500
**                                                                      *      003510
**  *****NJNPLO IS THE NUMBER OF JOBS NOT PERMITTED TO LOGIN            *      003520
**  *****SF.DISK IS THE DISK STATUS FLAG   0 = IDLE,   1 = BUSY         *      003530
**  *****JIS IS THE NUMBER OF JB INITIATORS AVAILABLE FOR BATCH JOBS    *      003540
**  *****JISI IS TH NUMBER OF JOB INITIATORS AVAIL FOR INTERACTIVE JOB  *      003550
**************************************************************************     003560
DEFINE I AS INTEGER VARIAPLE                                                   003570
LET SECOND = 1.0/60.0                                                          003580
LET MS = SECOND/1000.0                                                         003590
LET SEC.DAYS=(1.0/60.0)*(1.0/60.0)*(1.0/24.0)                                  003600
LET MIL.SEC.DAYS=(1.0/1000.0)*SEC.DAYS                                         003610
LET FF=0                                                                       003620
LET FF=1                                                                       003630
LET NBJ = 0                                                                    003640
LET NIJ = 0                                                                    003650
LET NJNPLO =0                                                                  003660
LET JIS = 6                                                                    003670
LET JISI = 35                                                                  003680
LET JN = 0                                                                     003690
LET CPU.SF=0                                                                   003700
LET SF.DISK = 0                                                                003710
LET STATUS.PRIORITY = 0                                                        003720
LET PJOBFLAG=0                                                                 003730
LET POUTFLAG=0                                                                 003740
LET PCMFLAG=0                                                                  003750
LET PFLAG = 0                                                                  003760
                                                                               003770
```

```
LET GAIN = 0.0                                                          003780
LET GAIN.FACTOR = 0.0                                                   003790
LET ETME = 0.0                                                          003800
LET N.JOBS.PROCESSED = 0                                                003810
LET LOAD.F=0                                                            003820
LET FLAG=1                                                              003830
LET FLAG = 0                                                            003840
LET WARM.UP.FLAG=0                                                      003850
LET W.UP.TIME=0.0                                                       003860
LET T.CPU.TIME=0.0                                                      003870
LET TCPUT=0.0                                                           003880
LET CU=0.0                                                              003890
LET CURRENT.UTILIZATION=0.0                                             003900
RESERVE MAIN.MEMORY AS 4000                                             003910
FOR B=1 TO 5, DO                                                        003920
LET ST=REAL.F(RANDI.F(5,60,1))*SEC.DAYS                                 003930
SCHEDULE A B.JOB.REQUEST IN ST DAYS                                     003940
LOOP                                                                    003950
LET BNUMBER =5                                                          003960
LET B.COUNTER =0                                                        003970
LET INUMBER=0                                                           003980
LET I.COUNTER=0                                                         003990
SCHEDULE A B.JOB.REQUEST NOW                                            004000
SCHEDULE A I.JOB.REQUEST NOW                                            004010
SCHEDULE A CPU.UTILIZATION IN 10.0*SECOND MINUTES                       004020
SCHEDULE A P.INCREASE IN SECOND MINUTES                                 004030
SCHEDULE A PCM.INCREASE IN SECOND MINUTES                               004040
SCHEDULE A P.OUTP.INCREASE IN SECOND MINUTES                            004050
SCHEDULE A PJQ.INCREASE IN SECOND MINUTES                               004060
SCHEDULE A CPU.ANALYSIS IN 1.0*SECOND MINUTES                           004070
SCHEDULE A DIAGNOSTIC IN .04 DAYS                                       004080
SCHEDULE A CEASE.SIM IN .162 DAYS                                       004090
PRINT 1 LINE THUS                                                       004100
EXIT MAIN                                                               004110
START SIMULATION                                                        004120
END                                                                     004130
```

```
EVENT DIAGNOSTIC                                                        004180
LET FLAG = 1                                                            004190
LET FLAG = 0                                                            004200
RETURN                                                                  004210
END                                                                     004220
EVENT CPU.ANALYSIS SAVING THE EVENT NOTICE                              004230
PRINT 1 LINE THUS                                                       004340
                              SECOND STATS                              004350
PRINT 1 LINE WITH JN AND TIME.V THUS                                    004360
                     ***.     *.********                                004370
     IF WARM.UP.FLAG GE 1                                               004380
     LET CURRENT.UTILIZATION=(T.CPU.TIME/(TIME.V-W.UP.TIME))*100.0      004390
     PRINT 1 LINE WITH CURRENT.UTILIZATION THUS                         004400
                                                          ***.**        004410
     ELSE                                                               004420
     LET CU=(TCPUT/TIME.V)*100.0                                        004430
     PRINT 1 LINE WITH CU THUS                                          004440
                                                          ***.**        004450
     ALWAYS                                                             004460
SCHEDULE THE CPU.ANALYSIS IN 1.0*SECOND MINUTES                         004470
IF TIME.V GT 30.0*SEC.DAYS                                              004480
CANCEL CPU.ANALYSIS                                                     004490
RETURN                                                                  004500
ELSE                                                                    004510
RETURN                                                                  004520
END                                                                     004530
                                                                        004540
EVENT CPU.UTILIZATION SAVING THE EVENT NOTICE                           004550
PRINT 1 LINE THUS                                                       004640
      CURRENT STATS                                                     004650
PRINT 1 LINE WITH JN AND TIME.V THUS                                    004660
***.     *.********                                                     004670
     IF FF=0                                                            004680
PRINT 1 LINE WITH T.CPU.TIME THUS                                       004690
     *.********                                                        004700
     ELSE                                                               004710
     ALWAYS                                                             004720
                                                                        004730
```

```
      IF WARM.UP.FLAG GE 1                                                    004740
      LET CURRENT.UTILIZATION=(T.CPU.TIME/(TIME.V-W.UP.TIME))*100.0           004750
      PRINT 1 LINE WITH CURRENT.UTILIZATION THUS                              004760
                              ***.**                                          004770
      ELSE                                                                    004780
      LET CU=(TCPUT/TIME.V)*100.0                                             004790
      PRINT 1 LINE WITH CU THUS                                               004800
                              ***.**                                          004810
      ALWAYS                                                                  004820
SCHEDULE THE CPU.UTILIZATION IN 10.0*SECOND MINUTES                           004830
IF TIME.V GT 500.0*SEC.DAYS                                                   004840
  CANCEL CPU.UTILIZATION                                                      004850
  RETURN                                                                      004860
ELSE                                                                          004870
RETURN                                                                        004880
END                                                                           004890
                                                                              004900
EVENT B.JOB.REQUEST SAVING THE EVENT NOTICE                                   005000
DEFINE PL AS REAL VARIABLE                                                    005010
DEFINE I.O AS REAL VARIABLE                                                   005020
DEFINE CP.T AS REAL VARIABLE                                                  005030
**************************************************************************  005040
**     NEEDED.RESOURCES.F IS A NUMERICAL INDICATION OF RESOURCES        *     005050
**                                                          NEEDED      *     005060
**     NBJ IS THE NUMBER OF BATCH JOBS IN THE SYSTEM                    *     005070
**                                                                      *     005080
**     JOB.TYPE 1=BATCH,2=INTERACTIVE                                   *     005090
**                                                                      *     005100
**     JOB.IO.REQUESTS IS THE TOTAL NUMBER OF I/O OPERATIONS            *     005110
**          REQUESTED BY A SINGLE JOB                                   *     005120
**                                                                      *     005130
**     IO.SERVICE.TIME IS I/O TIME REQUIRED TO PROCESS A SINGLE I/O     *     005140
**                                                              TASK    *     005150
**     XX.CPU.SERVICE.TIME IS THE CPU TIME REQUIRED TO PROCESS          *     005160
**          UNTIL AN I/O                                                *     005170
**                                                                      *     005180
**     JIS IS THE NUMBER OF JOB INITIATORS AVAILABLE                    *     005190
**                                                                      *     005200
```

```
**      MEMORY.REQUEST IS THE AMOUNT OF CORE NEEDED                          *
**                                                                           *
**      PAGE.LENGTH IS THE TOTAL PAGES IN THE OUTPUT                         *
**                                                                           *
**      T.LEFT.IN.CPU IS THE AMOUNT OF TIME LEFT UNTIL NEXT I/O              *
**      TM = TAPE.MOUNT = 0 MEANS JOB DOES NOT NEED TAPE MOUNTED             *
**                      = 1 MEANS JOB NEEDS TAPE MOUNTED                     *
**      DM = DISK.MOUNT = 0 MEANS JOB DOES NOT NEED DIS.MOUNTED              *
**                        1 MEANS JOB NEEDS DISK MOUNTED                     *
**                                                                           *
**                                          TAPE MOUNTS     DISC MOUNT       *
**  CLASS       REGION          TIME         PERMITTED       PERMITTED       *
**  A           K<=200          T<5          NO              NO              *
**  B           K<=250          T<60         NO              NO              *
**  C           250<K<=500      T<60         NO              NO              *
**  D           K<=200          T<=5         YES             NO              *
**  E           K<=250          T<=60        YES             NO              *
**  F           250<K<=500      T<=60        YES             NO              *
**  G           K<=200          T<=5         YES             YES             *
**  H           K<=250          T<=60        YES             YES             *
**  I           250<K<=500      T<=60        YES             YES             *
**  J           K<=500          T<=60        YES             NO              *
**  L           K<=500          T<=60        YES             YES             *
**WHERE K IS THE REGION SIZE IN BLOCKS OF 1024 BYTES AND T IS CPU             *
**TIME IN MINUTES.                                                           *
*************************************************************************
  IF TIME.V GT .004
  LET WARM.UP.FLAG=WARM.UP.FLAG+1
  ELSE
  ALWAYS
     IF WARM.UP.FLAG=1
     LET W.UP.TIME=TIME.V
     ELSE
     ALWAYS
IF NBJ GE JIS
SCHEDULE THE B.JOB.REQUEST IN 1 MINUTES
GO TO 'R'
ELSE
```

```
   IF B.COUNTER GE BNUMBER                                                  005590
   SCHEDULE THE B.JOB.REQUEST IN POISSON.F(.025,3) HOURS                    005600
   ELSE                                                                     005610
   LET B.COUNTER = B.COUNTER+1                                              005620
   ALWAYS                                                                   005630
CREATE A JOB CALLED AMIS                                                    005640
LET JN = JN +1                                                              005650
PRINT 1 LINE WITH JN AND TIME.V THUS                                        005660
     ***.    *.********                                                     005670
     LET JOBNUM(AMIS) = JN                                                  005680
 **  *****ESTABLISHES THIS IS A BATCH JOB                                   005690
LET JOB.TYPE(AMIS) = 1                                                      005700
LET NBJ = NBJ + 1                                                           005710
LET TLF(AMIS) = 0                                                           005720
**  *****ESTABLISHES BATCH CP.TIME                                          005730
LET CP.TIME(AMIS) =NORMAL.F(26.6,24.0,5)*SEC.DAYS                           005740
   IF CP.TIME(AMIS) GT 3590.0*SEC.DAYS                                      005750
   LET CP.TIME(AMIS) = 3590.0*SEC.DAYS                                      005760
   ELSE                                                                     005770
   ALWAYS                                                                   005780
      IF CP.TIME(AMIS) LT 10.0*SEC.DAYS                                     005790
      LET CP.TIME(AMIS)=10.0*SEC.DAYS                                       005800
      ELSE                                                                  005810
      ALWAYS                                                                005820
LET T = CP.TIME(AMIS)                                                       005830
LET DISK.MOUNT = RANDT.F(1,100,8)                                           005840
      IF DISK.MOUNT GT 85                                                   005850
      LET DISK.MOUNT = 0                                                    005860
      ELSE                                                                  005870
      LET DISK.MOUNT = 1                                                    005880
      ALWAYS                                                                005890
      LET TAPE.MOUNT = RANDI.F(1,100,7)                                     005900
      IF TAPE.MOUNT GT 70                                                   005910
      LET TAPE.MOUNT = 0                                                    005920
      ELSE                                                                  005930
      LET TAPE.MOUNT = 1                                                    005940
      ALWAYS                                                                005950
```

```
**  *****ESTABLISHES IO.TIME                                                005960
LET IO.TIME(AMIS) = NORMAL.F(40.0, 80.0,2)*SEC.DAYS                         005970
IF IO.TIME(AMIS) LE 40.0*SEC.DAYS                                           005980
LET IO.TIME(AMIS)=40.0*SEC.DAYS                                             005990
ELSE                                                                        006000
ALWAYS                                                                      006010
**  ***** ESTABLISHING CORE REQUIREMENTS                                    006020
LET MEMORY.REQUEST(AMIS) = RANDI.F(100,500,7)                               006030
LET K = MEMORY.REQUEST(AMIS)                                                006040
**  *****ESTABLISHES JOB RESOURCES NEEDED                                   006050
LET RE.SOURCES(AMIS) = 1                                                    006060
LET NEEDED.RESOURCES.F(AMIS) = RANDI.F(10,20,1)                             006070
LET P.OF.RESOURCES(AMIS) = NEEDED.RESOURCES.F(AMIS)                         006080
                                                                            006090
                                                                            006100
                                                                            006110
**  *****ESTABLISHES THE NUMBER OF I/O REQUESTS                             006120
LET JOB.IO.REQUESTS(AMIS)=NORMAL.F(900.0,90.0,3)                            006130
                  IF JOB.IO.REQUESTS(AMIS) LT 5                             006140
                  LET JOB.IO.REQUESTS(AMIS)=RANDI.F(5,30,7)                 006150
                  ELSE                                                      006160
                  ALWAYS                                                    006170
                                                                            006180
**  *****ESTABLISHES PAGE LENGTH                                            006190
LET PL = (JOB.IO.REQUESTS(AMIS)/1)*LOG.NORMAL.F(10.0,5.0,1)                 006200
LET PAGE.LENGTH(AMIS) = INT.F(PL) + 1                                       006210
                                                                            006220
**  *****ESTABLISHES IO.TIME FOR 1ST I/O TASK                               006230
LET I.O = 1/JOB.IO.REQUESTS(AMIS)                                           006240
LET IO.SERVICE.TIME(AMIS) = I.O * IO.TIME(AMIS)                             006250
                                                                            006260
**  *****ESTABLISHING THE PRIORITY OF THE JOB                               006270
**  *****ESTABLISHING CLASS PRIORITY                                        006280
  IF K LE 200 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0                  006290
  LET PRIORITY(AMIS) = RANDI.F(190,200,1)                                   006300
GO TO *BYPASS*                                                              006310
```

```
  ELSE                                                                                    006320
IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0                                  006330
LET PRIORITY(AMIS) = RANDI.F(180,189,2)                                                   006340
GO TO *BYPASS*                                                                            006350
ELSE                                                                                      006360
  IF K GT 260 AND K LE 500 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0                   006370
  LET PRIORITY(AMIS) = RANDI.F(170,179,3)                                                 006380
GO TO *BYPASS*                                                                            006390
  ELSE                                                                                    006400
IF K LE 200 AND T LE 300*SEC.DAYS AND TM = 1 AND DM = 0                                   006410
LET PRIORITY(AMIS) = RANDI.F(160,169,4)                                                   006420
GO TO *BYPASS*                                                                            006430
ELSE                                                                                      006440
  IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0                                006450
  LET PRIORITY(AMIS) = RANDI.F(150,159,5)                                                 006460
GO TO *BYPASS*                                                                            006470
  ELSE                                                                                    006480
IF K  GT 260 AND K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0                    006490
LET PRIORITY(AMIS) = RANDI.F(140,149,6)                                                   006500
GO TO *BYPASS*                                                                            006510
ELSE                                                                                      006520
  IF K LE 200 AND T LE 300*SEC.DAYS AND TM = 1 AND DM = 1                                 006530
  LET PRIORITY(AMIS) = RANDI.F(130,139,7)                                                 006540
GO TO *BYPASS*                                                                            006550
  ELSE                                                                                    006560
IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1                                  006570
LET PRIORITY(AMIS) = RANDI.F(120,129,8)                                                   006580
GO TO *BYPASS*                                                                            006590
ELSE                                                                                      006600
  IF K GT 260 AND    K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1                006610
  LET PRIORITY(AMIS) = RANDI.F(110,119,9)                                                 006620
GO TO *BYPASS*                                                                            006630
  ELSE                                                                                    006640
IF K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0                                  006650
LET PRIORITY(AMIS) = RANDI.F(100,109,1)                                                   006660
GO TO *BYPASS*                                                                            006670
ELSE
```

```
  IF K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1                    006680
LET PRIORITY(AMIS) = RANDI.F(90,99,2)                                         006690
  ELSE                                                                        006700
ALWAYS                                                                        006710
LET PRIORITY(AMIS)=80                                                         006720
'BYPASS'                                                                      006730
                                                                              006740
**  *****ESTABLISHES THE JOB STARTING TIME                                    006750
LET F.LOADER(AMIS)=0                                                          006760
LET ARRIVAL.TIME(AMIS) = TIME.V                                               006770
                                                                              006780
**  *****ESTABLISHING THE NEEDED CPU TIME UNTIL 1ST I/O                       006790
LET CP.T=1/JOB.IO.REQUESTS(AMIS)                                              006800
LET XX.CPU.SERVICE.TIME(AMIS) = CP.T * CP.TIME(AMIS)                          006810
LET XX..CPU.SERVICE.TIME(AMIS) = XX.CPU.SERVICE.TIME(AMIS)                    006820
    IF FF=0                                                                   006830
PRINT 1 LINE THUS                                                             006840
   BATCH STATS                                                                006850
PRINT 1 LINE WITH JOB.TYPE(AMIS), JOBNUM(AMIS), IO.SERVICE.TIME(AMIS),        006860
PRIORITY(AMIS), XX.CPU.SERVICE.TIME(AMIS), XX..CPU.SERVICE.TIME(AMIS),        006870
CP.TIME(AMIS), JOB.IO.REQUESTS(AMIS) AND TIME.V THUS                          006880
*.  ***.  .*******  ***.  .*******  .*******  .*******  ***.  .*******        006890
    ELSE                                                                      006900
    ALWAYS                                                                    006910
LET TOTAL.JOB.CPU.TIME=TOTAL.JOB.CPU.TIME+IO.TIME(AMIS)                       006920
LET TOT.JOB.IO.TIME=TOT.JOB.IO.TIME+IO.TIME(AMIS)                             006930
LET B.MEMORY.REQUEST = MEMORY.REQUEST(AMIS)                                   006940
LET NSUB = NBJ                                                                006950
LET B.CPU.TIME = CP.TIME(AMIS)                                                006960
LET BATCH.IO=IO.TIME(AMIS)                                                    006970
FILE AMIS IN THE JOB.QUEUE                                                    006980
SCHEDULE A REQUEST.CM NOW                                                     006990
'R'                                                                           007000
RETURN                                                                        007010
END                                                                           007020
                                                                              007030
```

```
                                                                        007040
EVENT P.INCREASE SAVING THE EVENT NOTICE                                007130
**********************************************************************  007140
**      PLAG = 0 MEANS NO JOB HAS PRIORITY INCREASE                   *  007150
**               1 MEANS A JOB HAS HAD ITS PRIORITY INCREASED         *  007160
**********************************************************************  007170
FOR EACH JOB IN THE CPU..REQUEST.QUEUE, WITH                            007180
PRIORITY = 5, FIND THE FIRST CASE                                       007190
IF NONE                                                                 007200
SCHEDULE THE P.INCREASE IN 10 HOURS                                     007210
RETURN                                                                  007220
ELSE                                                                    007230
IF PFLAG = 1                                                            007240
SCHEDULE THE P.INCREASE IN 10.0*MIL.SEC.DAYS DAYS                       007250
RETURN                                                                  007260
ELSE                                                                    007270
LET PFLAG = 1                                                           007280
REMOVE JOB FROM CPU..REQUEST.QUEUE                                      007290
LET PRIORITY(JOB) = 15                                                  007300
FILE JOB IN CPU..REQUEST.QUEUE                                          007310
REMOVE FIRST AMIS FROM CPU..REQUEST.QUEUE                               007320
FILE AMIS IN CPU..REQUEST.QUEUE                                         007330
SCHEDULE THE P.INCREASE IN SECOND MINUTES                               007340
RETURN                                                                  007350
END                                                                     007360
                                                                        007370
EVENT P.OUTP.INCREASE SAVING THE EVENT NOTICE                           007470
**********************************************************************  007480
**      POUTFLAG=1 MEANS NO JOB HAS ITS PRIORITY INCREASED            *  007490
**                 1 MEANS A JOB HAS ITS PRIORITY INCREASED           *  007500
**********************************************************************  007510
FOR EACH JOB IN THE AN.OUTPUT.QUEUE, WITH                               007520
PRIORITY = 5,FIND THE FIRST CASE                                        007530
IF NONE                                                                 007540
SCHEDULE THE P.OUTP.INCREASE IN 10 HOURS                                007550
RETURN                                                                  007560
ELSE                                                                    007570
```

```
IF POUTFLAG=1                                                          007580
SCHEDULE THE P.OUTP.INCREASE IN 10.0*MIL.SEC.DAYS DAYS                 007590
RETURN                                                                 007600
ELSE                                                                   007610
LET POUTFLAG=1                                                         007620
REMOVE JOB FROM AN.OUTPUT.QUEUE                                        007630
LET PRIORITY(JOB) = 15                                                 007640
FILE JOB IN AN.OUTPUT.QUEUE                                            007650
REMOVE FIRST AMIS FROM AN.OUTPUT.QUEUE                                 007660
FILE AMIS IN THE AN.OUTPUT.QUEUE                                       007670
SCHEDULE THE P.OUTP.INCREASE IN SECOND MINUTES                         007680
RETURN                                                                 007690
END                                                                    007700
                                                                       007710
**********************************************************************  007810
**     PJQFLAG = 0 MEANS NO JOB HAS HAD ITS PRIORITY INCREASED       *  007820
**               1 MEANS A JOB HAS ITS PRIORITY INCREASED            *  007830
**********************************************************************  007840
EVENT PJQ.INCREASE SAVING THE EVENT NOTICE                             007850
FOR EACH JOB IN THE JOB.QUEUE, WITH                                    007860
PRIORITY = 5,FIND THE FIRST CASE                                       007870
IF NONE                                                                007880
SCHEDULE THE PJQ.INCREASE IN 10 HOURS                                  007890
RETURN                                                                 007900
ELSE                                                                   007910
IF PJQFLAG = 1                                                         007920
SCHEDULE THE PJQ.INCREASE IN 10.0*MIL.SEC.DAYS DAYS                    007930
RETURN                                                                 007940
ELSE                                                                   007950
LET PJQFLAG=1                                                          007960
REMOVE JOB FROM JOB.QUEUE                                              007970
LET PRIORITY(JOB) = 15                                                 007980
FILE JOB IN JOB.QUEUE                                                  007990
REMOVE AMIS FROM JOB.QUEUE                                             008000
FILE AMIS IN JOB.QUEUE                                                 008010
SCHEDULE THE PJQ.INCREASE IN SECOND MINUTES                            008020
RETURN                                                                 008030
END                                                                    008040
```

AD-A055 209 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 5/1
SIMULATION OF THE ACQUISITION MANAGEMENT INFORMATION SYSTEM.(U)
MAR 77 A H LINDER

UNCLASSIFIED AFIT/GCS/MA/78-1 NL

2 OF 2
AD
A055209

END
DATE
FILMED
7-78
DDC

```
                                                                        008050
EVENT PCM.INCREASE SAVING THE EVENT NOTICE                              008140
**********************************************************************  008150
**     PCMFLAG = 0 MEANS NO JOB HAS ITS PRIORITY INCREASED          *   008160
**               1 MEANS A JOB HAS ITS PRIORITY INCREASED           *   008170
**********************************************************************  008180
FOR EACH JOB IN THE CM.QUEUE,WITH                                       008190
PRIORITY = 5,FIND THE FIRST CASE                                        008200
IF NONE                                                                 008210
SCHEDULE THE PCM.INCREASE IN 10 HOURS                                   008220
RETURN                                                                  008230
ELSE                                                                    008240
IF PCMFLAG = 1                                                          008250
SCHEDULE THE PCM.INCREASE IN 10.0*MIL.SEC.DAYS DAYS                     008260
RETURN                                                                  008270
ELSE                                                                    008280
LET PCMFLAG=1                                                           008290
REMOVE JOB FROM CM.QUEUE                                                008300
LET PRIORITY(JOB) = 15                                                  008310
FILE JOB IN CM.QUEUE                                                    008320
REMOVE FIRST AMIS FROM CM.QUEUE                                         008330
FILE AMIS IN CM.QUEUE                                                   008340
SCHEDULE THE PCM.INCREASE IN SECOND MINUTES                             008350
RETURN                                                                  008360
END                                                                     008370
                                                                        008380
EVENT I.JOB.REQUEST                                                     008470
DEFINE PL AS REAL VARIABLE                                              008480
DEFINE I.O AS REAL VARIABLE                                             008490
DEFINE CP.T AS REAL VARIABLE                                            008500
**********************************************************************  008510
**     JOB..IO.REQUESTS = TOTAL NUMBER I/O REQUESTS                 *   008520
**     JOB.IO.REQUETS = TOTAL NUMBER OF I/O REQUESTS PER CARRIAGE RET*  008530
**     NEEDED.RESOURCES.F IS A NUMERICAL INDICATION OF RESOURCES    *   008540
**                                                        NEEDED    *   008550
**                                                                  *   008560
**     NIJ IS THE NUMBER OF INTERACTIVE JOBS IN THE SYSTEM          *   008570
**                                                                  *
```

```
**      JOB.TYPE  1=BATCH, 2=INTERACTIVE                                  *
**                                                                        *
**      PAGE.LENGTH IS THE NUMBER OF OUTPUT PAGES                         *
**                                                                        *
**      IO.SERVICE.TIME IS THE I/O TIME REQUIRED TO PROCESS A SINGLE      *
**                                 I/O TASK                               *
**                                                                        *
**      MEMORY.REQUEST IS THE AMOUNT OF CORE NEEDED                       *
**                                                                        *
**                                          TAPE MOUNTS     DISC MOUNT    *
**  CLASS       REGION          TIME         PERMITTED       PERMITTED    *
**  A           K<=200          T<5          NO              NO           *
**  B           K<=260          T<60         NO              NO           *
**  C           260<K<=500      T<60         NO              NO           *
**  D           K<=200          T<=5         YES             NO           *
**  E           K<=260          T<=60        YES             NO           *
**  F           260<K<=500      T<=60        YES             NO           *
**  G           K<=200          T<=5         YES             YES          *
**  H           K<=260          T<=60        YES             YES          *
**  I           260<K<=500      T<=60        YES             YES          *
**  J           K<=500          T<=60        YES             NO           *
**  L           K<=500          T<=60        YES             YES          *
**WHERE K IS THE REGION SIZE IN BLOCKS OF 1024 BYTES AND T IS CPU          *
**TIME IN MINUTES.                                                         *
**      TM = TAPE.MOUNT = 0 MEANS JOB DOES NOT NEED TAPE MOUNTED          *
**                      = 1 MEANS JOB NEEDS TAPE MOUNTED                  *
**      DM = DISK.MOUNT = 0 MEANS JOB DOES NOT NEED DISK.MOUNTED          *
**                        1 MEANS JOB NEEDS DISK MOUNTED                  *
**                                                                        *
**      JOB.IO.REQUESTS IS THE TOTAL NUMBER OF I/O OPERATIONS REQUESTED   *
**           A SINGLE JOB                                                 *
**                                                                        *
**      XX.CPU.SERVICE.TIME IS THE CPU TIME REQUIRED TO PROCESS           *
**           UNTIL NEXT I/O                                               *
**************************************************************************
  IF TIME.V GT .014
  LET WARM.UP.FLAG=WARM.UP.FLAG+1
  ELSE
  ALWAYS
```

```
      IF WARM.UP.FLAG=1                                                    008980
      LET W.UP.TIME=TIME.V                                                 008990
      ELSE                                                                 009000
      ALWAYS                                                               009010
LET NIJ = NIJ + 1                                                          009020
      IF NIJ GT 35                                                         009030
      LET NIJ = NIJ-1                                                      009040
      LET NJNPLO = NJNPLO + 1                                              009050
      SCHEDULE A I.JOB.REQUEST IN 1 MINUTES                                009060
      RETURN                                                               009070
      ELSE                                                                 009080
  IF I.COUNTER GE INUMBER                                                  009090
  SCHEDULE A I.JOB.REQUEST IN POISSON.F(.1117,8) HOURS                     009100
  ELSE                                                                     009110
  LET I.COUNTER=I.COUNTER+1                                                009120
  ALWAYS                                                                   009130
CREATE A JOB CALLED AMIS                                                   009140
LET JN = JN +1                                                             009150
PRINT 1 LINE WITH JN AND TIME.V THUS                                       009160
      ***.    *.********                                                   009170
      LET JOBNUM(AMIS) = JN                                                009180
**  *****ESTABLISHES THIS IS A INTERACTIVE JOB                             009190
LET JOB.TYPE(AMIS) = 2                                                     009200
LET TLF(AMIS) = 0                                                          009210
**  *****ESTABLISHES INTERACTIVE CP.TIME                                   009220
LET CP.TIME(AMIS) =NORMAL.F(40.0,48.0,5)*SEC.DAYS                          009230
  IF CP.TIME(AMIS) GT 3590.0*SEC.DAYS                                      009240
  LET CP.TIME(AMIS) = 3590.0*SEC.DAYS                                      009250
  ELSE                                                                     009260
  ALWAYS                                                                   009270
      IF CP.TIME(AMIS) LT 15.0*SEC.DAYS                                    009280
      LET CP.TIME(AMIS)=15.0*SEC.DAYS                                      009290
      ELSE                                                                 009300
      ALWAYS                                                               009310
LET T = CP.TIME(AMIS)                                                      009320
LET DISK.MOUNT = RANDT.F(1,100,8)                                          009330
      IF DISK.MOUNT GT 85                                                  009340
      LET DISK.MOUNT = 0                                                   009350
      ELSE                                                                 009360
```

```
      LET DISK.MOUNT = 1                                                    009370
      ALWAYS                                                                009380
      LET TAPE.MOUNT = RANDI.F(1,100,7)                                     009390
      IF TAPE.MOUNT GT 70                                                   009400
      LET TAPE.MOUNT = 0                                                    009410
      ELSE                                                                  009420
      LET TAPE.MOUNT = 1                                                    009430
      ALWAYS                                                                009440
**    *****ESTABLISHES IO.TIME                                              009450
LET   IO.TIME(AMIS) = NORMAL.F(256.0, 64.0,2)*SEC.DAYS                      009460
      IF IO.TIME(AMIS) LT 50.0*SEC.DAYS                                     009470
      LET IO.TIME(AMIS)=UNIFORM.F(50.0,100.0,4)*SEC.DAYS                    009480
      ELSE                                                                  009490
      ALWAYS                                                                009500
**    *****ESTABLISHES CORE REQUIREMENTS                                    009510
LET   MEMORY.REQUEST(AMIS) = RANDI.F(100,500,7)                             009520
LET   K = MEMORY.REQUEST(AMIS)                                              009530
LET   RE.SOURCES(AMIS) = 1                                                  009540
**    *****ESTABLISHES JOB.RESOURCES NEEDED                                 009550
LET   NEEDED.RESOURCES.F(AMIS) = RANDI.F(10,20,1)                           009560
LET   P.OF.RESOURCES(AMIS) = NEEDED.RESOURCES.F(AMIS)                       009570
                                                                            009580
**    *****ESTABLISHES THE JOB STARTING TIME                                009590
LET   ARRIVAL.TIME(AMIS) = TIME.V                                           009600
LET   NO.CAR.RETURNS(AMIS) = RANDI.F(5,20,6)                                009610
**    *****ESTABLISHES PAGE LENGTH OF PUTPUT                                009620
LET   PL = LOG.NORMAL.F(20.0,2.0,9)                                         009630
LET   PAGE.LENGTH(AMIS) = INT.F(PL) + 1                                     009640
                                                                            009650
**    *****ESTABLISHES THE NUMBER OF I/O TASKS                              009660
LET   JOB..IO.REQUESTS(AMIS)=RANDI.F(2000,4000,9)                           009670
LET   JOB.IO.REQUESTS(AMIS)=INT.F(JOB..IO.REQUESTS(AMIS)/NO.CAR.RETURNS(AMIS))  009680
**    *****ESTABLISHES IO.TIME FOR 1ST I/O TASK                             009690
LET   I.O = 1/JOB..IO.REQUESTS(AMIS)                                        009700
LET   IO.SERVICE.TIME(AMIS) = I.O * IO.TIME(AMIS)                           009710
LET   JOB..IO.REQUESTS(AMIS)=JOB.IO.REQUESTS(AMIS)                          009720
**    *****ESTABLISHING PRIORITY OF THIS JOB                                009730
**    *****ESTABLISHING CLASS PRIORITY                                      009740
```

```
  IF K LE 200 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0                    009750
  LET PRIORITY(AMIS) = RANDI.F(190,200,1)                                     009760
GO TO *PASSBY*                                                                009770
  ELSE                                                                        009780
IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0                      009790
LET PRIORITY(AMIS) = RANDI.F(180,189,2)                                       009800
GO TO *PASSBY*                                                                009810
ELSE                                                                          009820
  IF K GT 260 AND K LE 500 AND T LE 3600*SEC.DAYS AND TM = 0 AND DM = 0       009830
  LET PRIORITY(AMIS) = RANDI.F(170,179,3)                                     009840
GO TO *PASSBY*                                                                009850
  ELSE                                                                        009860
IF K LE 200 AND T LE 300*SEC.DAYS AND TM = 1 AND DM = 0                       009870
LET PRIORITY(AMIS) = RANDI.F(160,169,4)                                       009880
GO TO *PASSBY*                                                                009890
ELSE                                                                          009900
  IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0                    009910
  LET PRIORITY(AMIS) = RANDI.F(150,159,5)                                     009920
GO TO *PASSBY*                                                                009930
  ELSE                                                                        009940
IF K  GT 260 AND K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0        009950
LET PRIORITY(AMIS) = RANDI.F(140,149,6)                                       009960
GO TO *PASSBY*                                                                009970
ELSE                                                                          009980
  IF K LE 200 AND T LE 300*SEC.DAYS AND TM = 1 AND DM = 1                     009990
  LET PRIORITY(AMIS) = RANDI.F(130,139,7)                                     010000
GO TO *PASSBY*                                                                010010
  ELSE                                                                        010020
IF K LE 260 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1                      010030
LET PRIORITY(AMIS) = RANDI.F(120,129,8)                                       010040
GO TO *PASSBY*                                                                010050
ELSE                                                                          010060
```

```
  IF K GT 260 AND    K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1
  LET PRIORITY(AMIS) = RANDI.F(110,119,9)
GO TO 'PASSBY'
  ELSE
IF K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 0
LET PRIORITY(AMIS) = RANDI.F(100,109,1)
GO TO 'PASSBY'
ELSE
  IF K LE 500 AND T LE 3600*SEC.DAYS AND TM = 1 AND DM = 1
LET PRIORITY(AMIS) = RANDI.F(90,99,2)
  ELSE
ALWAYS
LET PRIORITY(AMIS)=80
'PASSBY'
LET PRIORITY(AMIS) = PRIORITY(AMIS)+200
**  *****ESTABLISHING THE NEEDED CPU TIME UNTIL 1ST I/O TASK
LET CP.T=1/JOB..IO.REQUESTS(AMIS)
LET XX.CPU.SERVICE.TIME(AMIS) = CP.T * CP.TIME(AMIS)
LET XX..CPU.SERVICE.TIME(AMIS) = XX.CPU.SERVICE.TIME(AMIS)
F FLAG=1
PRINT 1 LINE THUS
            INTERACTIVE PARAMAETERS
LIST JOBNUM(AMIS)
LIST NO.CAR.RETURNS(AMIS)
LIST JOB.IO.REQUESTS(AMIS)
LIST JOB..IO.REQUESTS(AMIS)
LIST XX..CPU.SERVICE.TIME(AMIS)
LIST XX.CPU.SERVICE.TIME(AMIS)
LSE
LWAYS
```

```
     IF FF=0                                                                    010370
PRINT 1 LINE THUS                                                               010380
    INTERACTIVE STATS                                                           010390
PRINT 1 LINE WITH JOBNUM(AMIS), NO.CAR.RETURNS(AMIS), JOB.IO.REQUESTS(AMIS),    010400
JOB..IO.REQUESTS(AMIS), XX..CPU.SERVICE.TIME(AMIS), XX.CPU.SERVICE.TIME(AMIS),  010410
PRIORITY(AMIS), AND IO.TIME(AMIS) THUS                                          010420
***.    **.    **.    **.    .*******    .*******    ***.    .*******          010430
     ELSE                                                                       010440
     ALWAYS                                                                     010450
LET F.LOADER(AMIS)=0                                                            010460
LET TOTAL.JOB.CPU.TIME=TOTAL.JOB.CPU.TIME+CP.TIME(AMIS)                         010470
LET TOT.JOB.IO.TIME=TOT.JOB.IO.TIME+IO.TIME(AMIS)                               010480
     LET I.MEMORY.REQUEST = MEMORY.REQUEST(AMIS)                                010490
     LET NSUI = NIJ                                                             010500
     LET I.CPU.TIME = CP.TIME(AMIS)                                             010510
LET INTERACTIVE.IO = IO.TIME(AMIS)                                              010520
FILE AMIS IN THE JOB.QUEUE                                                      010530
SCHEDULE A REQUEST.CM IN 1.0 *MIL.SEC.DAYS DAYS                                 010540
RETURN                                                                          010550
END                                                                             010560
                                                                                010570
EVENT REQUEST.CM                                                                010580
**************************************************************************      010590
**     RE.SOURCES(AMIS) = 0 - ALL RESOURCES AVAILABLE                   *      010600
**                      = 1 - RESOURCES NOT AVAILABLE                   *      010610
**                                                                      *      010620
**     FLAG.C IS THE COUNTER USED TO DECREMENT CURRENT MEMORY.BEGINS    *      010630
**                                                                      *      010640
**     CONTIGOUS.SPACE IS THE AMOUNT OF CONTIGUOUS SPACE AVAILABLE      *      010650
**                                                                      *      010660
**     MEMORY.BLOCKS = MEMORY.REQUEST = CORE NEEDED                     *      010670
**                                                                      *      010680
**     MAIN.MEMORY IS THE CORE ARRAY                                    *      010690
**                                                                      *      010700
**     RESOURCES.F = 0 MEANS ALL RESOURCES REQUESTED ARE AVAILABLE      *      010710
**                 = 1 MEANS ALL RESOURCES NOT AVAILABLE                *      010720
**                                                                      *      010730
**************************************************************************      010740
```

```
IF THE JOB.QUEUE IS EMPTY                                                    010830
GO TO 'RE'                                                                   010840
ELSE                                                                         010850
REMOVE FIRST AMIS FROM JOB.QUEUE                                             010860
     LET PJQFLAG = 0                                                         010870
     IF RE.SOURCES(AMIS) = 0                                                 010880
     GO TO 'ENTER'                                                           010890
     ELSE                                                                    010900
CALL TRAFFIC.CONTROLLER GIVEN AMIS YIELDING RESOURCES.F                      010910
IF RESOURCES.F = 1                                                           010920
     IF PRIORITY(AMIS) = 15                                                  010930
     LET PRIORITY(AMIS) = 5                                                  010940
     ELSE                                                                    010950
     ALWAYS                                                                  010960
FILE AMIS IN WAIT.QUEUE                                                      010970
SCHEDULE A XX.TRAFFIC.CONTROLLER IN (1.0*MIL.SEC.DAYS) DAYS                  010980
RETURN                                                                       010990
ELSE                                                                         011000
'ENTER'                                                                      011010
LET FLAG.C = 0                                                               011020
LET CONTIGOUS.SPACE = 0                                                      011030
LET X = 1                                                                    011040
LET Y = 2                                                                    011050
LET MEMORY.BLOCKS = MEMORY.REQUEST(AMIS)                                     011060
**  *****MBC IS MEMORY NEEDED COUNTER                                        011070
**  ***** 0 IN MAIN.MEMORY MEANS THIS SECTION OF CORE IS AVAIBLE              011080
**  ***** 1 MEANS THIS SECTION OF CORE IS BEING USED                          011090
FOR MBC = 1 TO 4000, UNTIL X EQ  Y, DO                                       011100
IF MAIN.MEMORY(MBC) EQ 0                                                     011110
LET MEMORY.BEGINS=MBC - FLAG.C                                               011120
LET CONTIGOUS.SPACE = CONTIGOUS.SPACE+1                                      011130
LET FLAG.C=FLAG.C+1                                                          011140
ELSE                                                                         011150
LET CONTIGOUS.SPACE=0                                                        011160
LET FLAG.C=0                                                                 011170
GO TO 'RET'                                                                  011180
ALWAYS                                                                       011190
```

```
IF CONTIGOUS.SPACE GE MEMORY.BLOCKS                                          011200
LET START.MAIN.MEMORY = MEMORY.BEGINS                                        011210
LET SMM=START.MAIN.MEMORY                                                    011220
LET FINISH.MAIN.MEMORY= MEMORY.BEGINS + MEMORY.BLOCKS -1                     011230
LET FMM = FINISH.MAIN.MEMORY                                                 011240
LET START.MEMORY(AMIS)=START.MAIN.MEMORY                                     011250
LET END.MEMORY(AMIS) = FINISH.MAIN.MEMORY                                    011260
LET Y = 1                                                                    011270
FOR J=SMM TO FMM, DO                                                         011280
LET MAIN.MEMORY(J) =1                                                        011290
LOOP                                                                         011300
LET JOB.SIZE = (FMM-SMM)+1                                                   011310
ELSE                                                                         011320
LET Y = 2                                                                    011330
ALWAYS                                                                       011340
'RET'                                                                        011350
LOOP                                                                         011360
IF CONTIGOUS.SPACE LT MEMORY.BLOCKS                                          011370
**  *****THERE ISN'T ENOUGH CM AVAILABLE                                     011380
IF PRIORITY(AMIS) LE 15                                                      011390
LET PRIORITY(AMIS) = 5                                                       011400
LET PCMFLAG=0                                                                011410
ELSE                                                                         011420
ALWAYS                                                                       011430
FILE AMIS IN THE CM.QUEUE                                                    011440
SCHEDULE A   REQUEST.CM IN (1.0*MIL.SEC.DAYS) DAYS                           011450
ELSE                                                                         011460
CALL LOADER GIVEN JOB.SIZE, AMIS                                             011470
**  *****XXTIME.E.CPU.QUEUE IS THE TIME THE JOB ENTERED THE CPU.QUEUE        011480
LET XXTIME.E.CPU.QUEUE(AMIS)=TIME.V                                          011490
FILE AMIS IN THE CPU..REQUEST.QUEUE                                          011500
IF FLAG = 1                                                                  011510
                    IF JOBNUM(AMIS) EQ 2                                     011520
                    LIST TIME.V                                              011530
                    LIST STATUS.PRIORITY                                     011540
                    PRINT 1 LINE THUS                                        011550
                    EVENT REQUEST.CM                                         011560
                    ELSE                                                     011570
                    ALWAYS                                                   011580
ELSE
ALWAYS
```

```
                                                                        011600
SCHEDULE A CPU.REQUEST NOW                                              011610
ALWAYS                                                                  011620
'RE'                                                                    011630
RETURN                                                                  011640
END                                                                     011650
                                                                        011660
ROUTINE TRAFFIC.CONTROLLER GIVEN AMIS YIELDING RESOURCES.F              011760
**********************************************************************  011770
**     RESOURCES.F = 0 MEANS ALL RESOURCES REQUESTED ARE AVAILABLE    *  011780
**                 = 1 MEANS ALL RESOURCES NOT AVAILABLE              *  011790
**********************************************************************  011800
DEFINE UPPER AS INTEGER VARIABLE                                        011810
DEFINE RE.F AS INTEGER VARIABLE                                         011820
LET UPPER = P.OF.RESOURCES(AMIS)                                        011830
LET RE.F = RANDI.F(1,UPPER,3)                                           011840
IF RE.F LE 8                                                            011850
LET RESOURCES.F = 0                                                     011860
ELSE                                                                    011870
LET RESOURCES.F = 1                                                     011880
LET P.OF.RESOURCES(AMIS) = P.OF.RESOURCES(AMIS) - 2                     011890
ALWAYS                                                                  011900
RETURN                                                                  011910
END                                                                     011920
                                                                        011930
EVENT XX.TRAFFIC.CONTROLLER                                             012030
**********************************************************************  012040
**     RE.SOURCES(AMIS) = 0 - ALL RESOURCES AVAILABLE                 *  012050
**                      = 1 - RESOURCES NOT AVAILABLE                 *  012060
**********************************************************************  012070
REMOVE FIRST AMIS FROM WAIT.QUEUE                                       012080
CALL TRAFFIC.CONTROLLER GIVEN AMIS YIELDING RESOURCES.F                 012090
IF RESOURCES.F = 1                                                      012100
FILE AMIS IN WAIT.QUEUE                                                 012110
SCHEDULE A   XX.TRAFFIC.CONTROLLER IN (1.0*MIL.SEC.DAYS) DAYS           012120
ELSE                                                                    012130
LET RE.SOURCES(AMIS) = 0                                                012140
FILE AMIS IN JOB.QUEUE                                                  012150
SCHEDULE A REQUEST.CM NOW                                               012160
ALWAYS                                                                  012170
RETURN                                                                  012180
END                                                                     012190
```

```
ROUTINE LOADER GIVEN JOB.SIZE, AMIS
*****************************************************************
**      F.LOADER = 1 MEANS JOB NEEDS TO BE LOADED INTO MEMORY     *
**      F.LOADER = 0 MEANS JOB NEED NOT BE LOADED INTO MEMORY     *
**      LOAD.T IS THE TIME NECESSARY TO LOAD JOB INTO MEMORY      *
*****************************************************************

LET LOADING.T(AMIS) = (1.0/1000.0)*SEC.DAYS
LET F.LOADER (AMIS) = 1
RETURN
END

EVENT CPU.REQUEST SAVING THE EVENT NOTICE
*****************************************************************
**      CPU.SF IS THE CPU STATUS FLAG 1=BUSY,0=IDLE               *
**                                                                *
**      F.LOADER IS THE FLAG LOADER                               *
**            1 MEANS JOB NEEDS TO BE LOADED INTO MEMORY          *
**            0 MEANS JOB NEEDS CPU FOR ACTUAL PROCESSING         *
**                                                                *
**      LOAD.F =0 MEANS THE JOB DOES NOT NEED CPU FOR LOADING INTO MEM*
**        =1 MEANS JOB MUST BE LOADED INTO MEMORY                 *
**                                                                *
**      BLOCKED.TIME IS THE TOTAL TIME IN THE CPU.QUEUE           *
*****************************************************************
IF FLAG = 1
LIST CPU.SF
LIST LOAD.F
ELSE
ALWAYS
     IF LOAD.F=1
     SCHEDULE THE CPU.REQUEST IN (6.0/1000.0)*SEC.DAYS DAYS
     RETURN
     ELSE
IF THE CPU..REQUEST.QUEUE IS EMPTY
```

```
IF FLAG = 1                                                                    012740
PRINT 1 LINE THUS                                                              012750
CPU..REQUEST QUEUE IS EMPTY                                                    012760
ELSE                                                                           012770
ALWAYS                                                                         012780
DESTROY CPU.REQUEST                                                            012790
RETURN                                                                         012800
ELSE                                                                           012810
IF FLAG = 1                                                                    012820
PRINT 1 LINE THUS                                                              012830
CPU..REQUEST.QUEUE IS NOT EMPTY                                                012840
ELSE                                                                           012850
ALWAYS                                                                         012860
  FOR EACH AMIS IN CPU..REQUEST.QUEUE, WITH                                    012870
  F.LOADER(AMIS) EQ 1                                                          012880
  FIND THE FIRST CASE                                                          012890
  IF NONE  GO TO 'PROCESS'                                                     012900
  ELSE                                                                         012910
66          IF CPU.SF=0                                                        012920
            LET LOAD.F=1                                                       012930
            GO TO 'CKS'                                                        012940
            ELSE                                                               012950
            DESTROY CPU.REQUEST                                                012960
            RETURN                                                             012970
'PROCESS'                                                                      012980
IF FLAG = 1                                                                    012990
PRINT 1 LINE THUS                                                              013000
     BBBB                                                                      013010
ELSE                                                                           013020
ALWAYS                                                                         013030
FOR EACH AMIS IN CPU..REQUEST.QUEUE, WITH                                      013040
  F.LOADER(AMIS)=0 AND PRIORITY(AMIS) GT STATUS.PRIORITY,                      013050
  FIND THE FIRST CASE                                                          013060
  IF NONE                                                                      013070
IF FLAG = 1                                                                    013080
PRINT 1 LINE THUS                                                              013090
       THERE ARE NO JOBS THAT MEET THIS REQUIREMENT                            013100
```

```
ELSE                                                                    013110
ALWAYS                                                                  013120
DESTROY CPU.REQUEST                                                     013130
RETURN                                                                  013140
  ELSE                                                                  013150
IF FLAG = 1                                                             013160
PRINT 1 LINE THUS                                                       013170
      THIS JOB MADE IT PAST THE PRIORITY TEST                           013180
ELSE                                                                    013190
ALWAYS                                                                  013200
     IF CPU.SF = 1                                                      013210
     GO TO 'CHECK'                                                      013220
     ELSE                                                               013230
     CANCEL RELEASE.CPU                                                 013240
     DESTROY RELEASE.CPU                                                013250
                  IF FLAG = 1                                           013260
                 PRINT 1 LINE THUS                                      013270
                       AAAA                                             013280
                  ELSE                                                  013290
                  ALWAYS                                                013300
REMOVE FIRST ANDY FROM ACPU                                             013310
                 IF FLAG = 1                                            013320
                  PRINT 1 LINE THUS                                     013330
                  IN CPU.REQUEST SECTION A                              013340
                  LIST JOBNUM(ANDY)                                     013350
                  LIST TIME.V                                           013360
                  LIST STATUS.PRIORITY                                  013370
                  LIST PRIORITY(ANDY)                                   013380
                 ELSE                                                   013390
     ,           ALWAYS                                                 013400
     IF FF=0                                                            013410
PRINT 1 LINE THUS                                                       013420
   STATS ON ANDY JOBS                                                   013430
PRINT 1 LINE WITH TIME.V THUS                                           013440
                                                          .********     013450
PRINT 1 LINE WITH JOB.TYPE(ANDY), JOBNUM(ANDY), T.E.CPU(ANDY),          013460
AND XX.CPU.SERVICE.TIME(ANDY) THUS                                      013470
*.    ***.    .********    .********                                    013480
PRINT 1 LINE WITH T.CPU.TIME THUS                                       013490
*.********                                                             013500
```

```
      ELSE                                                                    013510
      ALWAYS                                                                  013520
  LET XX.CPU.SERVICE.TIME(ANDY)=XX.CPU.SERVICE.TIME(ANDY)                     013530
                               -(TIME.V-T.E.CPU(ANDY))                        013540
                               -IO.OVERHEAD.T                                 013550
IF XX.CPU.SERVICE.TIME(ANDY) LE 0.0                                           013560
LET XX.CPU.SERVICE.TIME(ANDY)=0.0                                             013570
ELSE                                                                          013580
ALWAYS                                                                        013590
LET XXTIME.E.CPU.QUEUE(ANDY)=TIME.V                                           013600
    IF WARM.UP.FLAG GE 1                                                      013610
  LET T.CPU.TIME=T.CPU.TIME+((TIME.V-T.E.CPU(ANDY))-IO.OVERHEAD.T)            013620
  LET GAIN=GAIN+((TIME.V-T.E.CPU(ANDY))-IO.OVERHEAD.T)                        013630
  ELSE                                                                        013640
  LET TCPUT = TCPUT + (TIME.V-T.E.CPU(ANDY) - IO.OVERHEAD.T)                  013650
  ALWAYS                                                                      013660
                  FILE ANDY IN CPU..REQUEST.QUEUE                             013670
                  REMOVE AMIS FROM CPU..REQUEST.QUEUE                         013680
      GO TO 'C'                                                               013690
'CKS'                                                                         013700
IF FLAG = 1                                                                   013710
PRINT 1 LINE THUS                                                             013720
JOB AT CKS                                                                    013730
ELSE                                                                          013740
ALWAYS                                                                        013750
LET CPU.SF = 1                                                                013760
REMOVE AMIS FROM CPU..REQUEST.QUEUE                                           013770
FILE AMIS IN ACPU                                                             013780
GO TO 'SCH'                                                                   013790
'CHECK'                                                                       013800
LET CPU.SF=1                                                                  013810
IF FLAG = 1                                                                   013820
PRINT 1 LINE THUS                                                             013830
        JOB MADE IT TO CHECK                                                  013840
ELSE                                                                          013850
ALWAYS                                                                        013860
REMOVE AMIS FROM  CPU..REQUEST.QUEUE                                          013870
```

```
'C'                                                                    013880
**  *****BLOCKED.TIME IS THE TOTAL TIME IN THE CPU QUEUE               013890
  IF WARM.UP.FLAG GE 1                                                 013900
  LET BLOCKED.TIME(AMIS) = TIME.V - XXTIME.E.CPU.QUEUE(AMIS) +         013910
  BLOCKED.TIME(AMIS)                                                   013920
  ELSE                                                                 013930
  ALWAYS                                                               013940
LET XXTIME.E.CPU.QUEUE(AMIS)=0.0                                       013950
LET STATUS.PRIORITY=PRIORITY(AMIS)                                     013960
FILE AMIS IN ACPU                                                      013970
'SCH'                                                                  013980
SCHEDULE A CPU.PROCESSING NOW                                          013990
DESTROY CPU.REQUEST                                                    014000
RETURN                                                                 014010
END                                                                    014020
                                                                       014030
EVENT CPU.PROCESSING                                                   014130
**********************************************************************  014140
**     CP.T.QUANTUM IS THE CPU TIME QUANTUM                          *  014150
**                                                                   *  014160
**     T.LEFT.IN.CPU IS THE REMAINING TIME UNTIL NEXT I/O            *  014170
**     TLF IS THE TIME LEFT  FOR I/O TASK (TIME LEFT FLAG)           *  014180
**          0 MEANS NO MORE CPU TIME NEEDED                          *  014190
**          1 MEANS MORE TIME NEEDED UNTIL NEXT I/O                  *  014200
**                                                                   *  014210
**     TIME.FLAG = 1 MEANS CPU.R EQUAL TO XX.CPU.SERVICE.TIME        *  014220
**          1 MEANS THE CPU TASK HAS BEEN COMPLETED                  *  014230
**          0 MEANS THAT THE CPU TASK NEEDS MORE TIME                *  014240
**                                                                   *  014250
**     IO.SERVICE.TIME IS THE IO.TIME REQUIRED TO PROCESS A SINGLE   *  014260
**          I/O TASK                                                 *  014270
**                                                                   *  014280
**     XX.CPU.SERVICE.TIME IS THE TIME NEEDED UNTIL NEXT I/O         *  014290
**                                                                   *  014300
**     CPU.R IS THE TIME TO RELEASE THE CPU                          *  014310
**                                                                   *  014320
```

```
**     LOAD.F = 0 MEANS THE JOB DOES NOT NEED LOADING INTO MEMORY    *
**              1 MEANS JOB MUST BE LOADED INTO MEMORY               *
**     TIME.E.CPU.QUEUE IS THE TIME THE JOB MOST RECENTLY ENTERED    *
**     THE CPU QUEUE                                                *
**     JOB.TYPE 1=BATCH, 2 = INTERACTIVE                            *
**************************************************************************
DEFINE TIME.FLAG AS INTEGER VARIABLE
DEFINE IO.OVERHEAD.T AS REAL VARIABLE
LET IO.OVERHEAD.T = (1.0/100.0)*SEC.DAYS
     FOR EACH AMIS IN ACPU, WITH
     F.LOADER(AMIS) EQ 1
     FIND THE FIRST CASE
     IF NONE GO TO 'A'
     ELSE
     REMOVE AMIS FROM ACPU
     FILE AMIS IN ACPU
     SCHEDULE A RELEASE.CPU IN (5.0/1000.0)*SEC.DAYS DAYS
     RETURN
'A'
REMOVE FIRST AMIS FROM ACPU
IF FLAG = 1
                    LIST TLF(AMIS)
                    LIST TIME.V
                    LIST PRIORITY(AMIS)
                    LIST JOBNUM(AMIS)
                    LIST STATUS.PRIORITY
                    PRINT 1 LINE THUS
                    EVENT CPU.PROCESSING
ELSE
ALWAYS
     IF TLF(AMIS) EQ 1
     LET XX.CPU.SERVICE.TIME(AMIS) = T.LEFT.IN.CPU(AMIS)
     ELSE
     ALWAYS
IF JOB.TYPE(AMIS) = 1
LET CP.T.QUANTUM=4000.0*SEC.DAYS
```

```
LET CPU.R = MIN.F(XX.CPU.SERVICE.TIME(AMIS),CP.T.QUANTUM)                    014690
     IF CP.T.QUANTUM EQ CPU.R                                                014700
     LET T.LEFT.IN.CPU(AMIS) = XX.CPU.SERVICE.TIME(AMIS) -CP.T.QUANTUM       014710
     LET TLF(AMIS)=1                                                         014720
     LET TIME.FLAG = 0                                                       014730
     ELSE                                                                    014740
     LET T.LEFT.IN.CPU(AMIS) = 0.0                                           014750
     LET TIME.FLAG = 1                                                       014760
     LET TLF(AMIS) = 0                                                       014770
     ALWAYS                                                                  014780
ELSE                                                                         014790
LET CP.T.QUANTUM=4000.0*SEC.DAYS                                             014800
LET CPU.R = MIN.F(XX.CPU.SERVICE.TIME(AMIS),CP.T.QUANTUM)                    014810
     IF CP.T.QUANTUM EQ CPU.R                                                014820
     LET T.LEFT.IN.CPU(AMIS) = XX.CPU.SERVICE.TIME(AMIS) -CP.T.QUANTUM       014830
     LET TLF(AMIS)=1                                                         014840
     LET TIME.FLAG = 0                                                       014850
     ELSE                                                                    014860
     LET TIME.FLAG = 1                                                       014870
     LET TLF(AMIS) = 0                                                       014880
     LET T.LEFT.IN.CPU(AMIS) = 0.0                                           014890
     ALWAYS                                                                  014900
ALWAYS                                                                       014910
     IF TIME.FLAG EQ 1                                                       014920
     SCHEDULE A RELEASE.CPU IN (CPU.R + IO.OVERHEAD.T) DAYS                  014930
     LET TS=CPU.R+IO.OVERHEAD.T                                              014940
     LET CPU.R=0.0                                                           014950
     LET TLF(AMIS) = 0                                                       014960
     ELSE                                                                    014970
     SCHEDULE THE RELEASE.CPU IN CPU.R DAYS                                  014980
     LET TS=0.0                                                              014990
     ALWAYS                                                                  015000
LET T.E.CPU(AMIS)=TIME.V                                                     015010
     IF FF=0                                                                 015020
PRINT 1 LINE THUS                                                            015030
   STATS IN CPU PROCESSING                                                   015040
PRINT 1 LINE WITH TS,CPU.R, JOBNUM(AMIS), JOB.TYPE(AMIS), PRIORITY(AMIS),    015050
XX.CPU.SERVICE.TIME(AMIS), XX..CPU.SERVICE.TIME(AMIS) AND TIME.V THUS        015060
.********  .********  ***.  *.  ***.  .********  .********  .*******         015070
```

```
PRINT 1 LINE WITH T.CPU.TIME THUS                                        015080
*.********                                                               015090
     ELSE                                                                015100
     ALWAYS                                                              015110
SCHEDULE A CPU.REQUEST NOW                                               015120
FILE AMIS IN ACPU                                                        015130
RETURN                                                                   015140
END                                                                      015150
                                                                         015160
EVENT RELEASE.CPU                                                        015260
**************************************************************************015270
**                                                                      *015280
**     JOB.IO.REQUESTS IS THE TOTAL NUMBER OF I/O OPERATIONS            *015290
**          REQUESTED BY A SINGLE BATCH JOB                             *015300
**          REQUESTED BY A SINGLE INTERACTIVE JOB PER CARRIAGE RETURN   *015310
**                                                                      *015320
**     CPU.SF IS THE CPU STATUS FLAG 0=IDLE, 1=BUSY                     *015330
**                                                                      *015340
**     T.LEFT.IN.CPU IS THE REMAINING TIME TO I/O                       *015350
**                                                                      *015360
**                                                                      *015370
**     PRIORITY 10 = INTERACTIVE                                        *015380
**              5 = BATCH                                               *015390
**                                                                      *015400
**     TLF=0 MEANS THE CPU TASK HAS BEEN COMPLETED                      *015410
**         1 MEANS THE CPU TASK NEEDS MORE TIME                         *015420
**                                                                      *015430
**     JOB.TYPE   1=BATCH, 2=INTERACTIVE                                *015440
**     XXTIME.E.CPU.QUEUE IS THE TIME THE JOB MOST RECENTLY ENTERED     *015450
**     THE CPU QUEUE                                                    *015460
**************************************************************************015470
     IF ACPU IS EMPTY                                                    015480
     RETURN                                                              015490
     ELSE                                                                015500
     DEFINE THINK.TIME AS INTEGER VARIABLE                               015510
     FOR EACH AMIS IN ACPU, WITH                                         015520
     E.LOADER(AMIS) EQ 1                                                 015530
     FIND THE FIRST CASE                                                 015540
     IF NONE GO TO '3'                                                   015550
```

```
     ELSE                                                                    015560
     REMOVE AMIS FROM ACPU                                                   015570
     FILE AMIS IN LOAD.QUEUE                                                 015580
     SCHEDULE A LOAD.DELAY IN (5.0/1000.0)*SEC.DAYS DAYS                     015590
     RETURN                                                                  015600
*B*                                                                          015610
LET CPU.SF=0                                                                 015620
REMOVE FIRST AMIS FROM ACPU                                                  015630
     LET JOB.IO.REQUESTS(AMIS) = JOB.IO.REQUESTS(AMIS) -1                    015640
IF FLAG = 1                                                                  015650
                    PRINT 1 LINE THUS                                        015660
                    RELEASE CPU SECTION A                                    015670
                    LIST JOBNUM(AMIS)                                        015680
                    LIST TIME.V                                              015690
          LIST TLF(AMIS)                                                     015700
                    LIST PRIORITY(AMIS)                                      015710
                              LIST STATUS.PRIORITY                           015720
                    PRINT 1 LINE THUS                                        015730
                    EVENT RELEASE.CPU                                        015740
ELSE                                                                         015750
ALWAYS                                                                       015760
     IF FF=0                                                                 015770
PRINT 1 LINE THUS                                                            015780
  STATS IN RELEASE CPU                                                       015790
PRINT 1 LINE WITH TIME.V THUS                                                015800
                                                          .********          015810
PRINT 1 LINE WITH JOBNUM(AMIS), JOB.TYPE(AMIS) AND                           015820
XX.CPU.SERVICE.TIME(AMIS) THUS                                               015830
***.   *.   .*******                                                         015840
PRINT 1 LINE WITH T.CPU.TIME THUS                                            015850
*.********                                                                   015860
     ELSE                                                                    015870
     ALWAYS                                                                  015880
IF WARM.UP.FLAG GE 1                                                         015890
LET GAIN=GAIN + (XX.CPU.SERVICE.TIME(AMIS)-IO.OVERHEAD.T)                    015900
LET T.CPU.TIME=T.CPU.TIME+(XX.CPU.SERVICE.TIME(AMIS)-IO.OVERHEAD.T)          015910
ELSE                                                                         015920
```

```
LET TCPUT = TCPUT + (XX.CPU.SERVICE.TIME(AMIS) - IO.OVERHEAD.T)                 015930
ALWAYS                                                                          015940
     IF TIME.V LT .002332                                                       015950
     LET FF=0                                                                   015960
     LET FF=1                                                                   015970
     ELSE                                                                       015980
     LET FF=0                                                                   015990
     LET FF=1                                                                   016000
     ALWAYS                                                                     016010
  IF JOB.TYPE(AMIS) EQ 1                                                        016020
  GO TO 'E'                                                                     016030
  ELSE                                                                          016040
                    IF FLAG = 1                                                 016050
PRINT 1 LINE THUS                                                               016060
RELEASE CPU SECTION B                                                           016070
LIST JOBNUM(AMIS)                                                               016080
LIST NO.CAR.RETURNS(AMIS)                                                       016090
LIST JOB.IO.REQUESTS(AMIS)                                                      016100
LIST XX..CPU.SERVICE.TIME(AMIS)                                                 016110
LIST STATUS.PRIORITY                                                            016120
LIST PRIORITY(AMIS)                                                             016130
LIST TLE(AMIS)                                                                  016140
LIST JOB..IO.REQUESTS(AMIS)                                                     016150
                    ELSE                                                        016160
                    ALWAYS                                                      016170
                    IF FLAG = 1                                                 016180
         IF NO.CAR.RETURNS(AMIS) LE 0                                           016190
         LIST JOBNUM(AMIS)                                                      016200
         LIST NO.CAR.RETURNS(AMIS)                                              016210
         LIST JOB.IO.REQUESTS(AMIS)                                             016220
         LIST JOB..IO.REQUESTS(AMIS)                                            016230
         LIST XX.CPU.SERVICE.TIME(AMIS)                                         016240
         LIST XX..CPU.SERVICE.TIME(AMIS)                                        016250
         ELSE                                                                   016260
         ALWAYS                                                                 016270
                    ELSE                                                        016280
                    ALWAYS                                                      016290
```

```
      IF NO.CAR.RETURNS(AMIS) LE 0                                                    016300
      GO TO 'EE'                                                                      016310
      ELSE                                                                            016320
        IF JOB.IO.REQUESTS(AMIS) LE 0                                                 016330
        LET NO.CAR.RETURNS(AMIS) = NO.CAR.RETURNS(AMIS) -1                            016340
        LET JOB.IO.REQUESTS(AMIS)=JOB..IO.REQUESTS(AMIS)                              016350
        LET XX.CPU.SERVICE.TIME(AMIS) = XX..CPU.SERVICE.TIME(AMIS)                    016360
        LET THINK.TIME = RANDI.F(2,15,3)                                              016370
        LET SPECIAL.CASE = RANDI.F(1,20,3)                                            016380
          IF SPECIAL.CASE GT 12                                                       016390
          LET THINK.TIME = 240                                                        016400
          ELSE                                                                        016410
          ALWAYS                                                                      016420
    IF WARM.UP.FLAG GE 1                                                              016430
    LET USER.RESPONSE.TIME=THINK.TIME                                                 016440
    ELSE                                                                              016450
    ALWAYS                                                                            016460
         CREATE A RESPONSE                                                            016470
         LET RNAME(RESPONSE) = AMIS                                                   016480
         LET STATUS.PRIORITY=0                                                        016490
         SCHEDULE THE RESPONSE IN (THINK.TIME*SECOND) MINUTES                         016500
         SCHEDULE A CPU.REQUEST NOW                                                   016510
         RETURN                                                                       016520
         ELSE                                                                         016530
    IF TLF(AMIS) EQ 0                                                                 016540
    GO TO 'REESTABLISH'                                                               016550
    ELSE                                                                              016560
    GO TO 'FILE'                                                                      016570
  'E'                                                                                 016580
  IF FLAG = 1                                                                         016590
  LIST JOB.IO.REQUESTS(AMIS)                                                          016600
  LIST TLF(AMIS)                                                                      016610
  LIST CPU.SF                                                                         016620
  ELSE                                                                                016630
  ALWAYS                                                                              016640
  IF JOB.IO.REQUESTS(AMIS) EQ 0                                                       016650
  IF PRIORITY (AMIS) LE 15                                                            016660
  LET PFLAG =0                                                                        016670
  LET PRIORITY(AMIS) = 5                                                              016680
  ELSE                                                                                016690
  ALWAYS                                                                              016700
```

```
LET STATUS.PRIORITY=0                                                    016710
FILE AMIS IN XX.BUFFER.QUEUE                                             016720
SCHEDULE A CPU.REQUEST IN 5.0*MIL.SEC.DAYS DAYS                          016730
SCHEDULE A IS.BUFFER.EMPTY NOW                                           016740
RETURN                                                                   016750
ELSE                                                                     016760
IF TLF(AMIS) EQ 0                                                        016770
IF PRIORITY (AMIS) LE 15                                                 016780
LET PFLAG =0                                                             016790
LET PRIORITY(AMIS) = 5                                                   016800
ELSE                                                                     016810
ALWAYS                                                                   016820
LET XX.CPU.SERVICE.TIME(AMIS) = XX..CPU.SERVICE.TIME(AMIS)               016830
CREATE A IO.DELAY                                                        016840
LET NAME(IO.DELAY) = AMIS                                                016850
LET STATUS.PRIORITY=0                                                    016860
IF FLAG = 1                                                              016870
                    LIST IO.SERVICE.TIME(AMIS)                           016880
ELSE                                                                     016890
ALWAYS                                                                   016900
LET GAIN=GAIN + IO.SERVICE.TIME(AMIS)                                    016910
SCHEDULE THE IO.DELAY IN IO.SERVICE.TIME(AMIS) DAYS                      016920
SCHEDULE A CPU.REQUEST NOW                                               016930
RETURN                                                                   016940
ELSE                                                                     016950
IF PRIORITY (AMIS) LE 15                                                 016960
LET PFLAG =0                                                             016970
LET PRIORITY(AMIS) = 5                                                   016980
ELSE                                                                     016990
ALWAYS                                                                   017000
FILE AMIS IN CPU..REQUEST.QUEUE                                          017010
LET STATUS.PRIORITY=0                                                    017020
SCHEDULE A CPU.REQUEST NOW                                               017030
RETURN                                                                   017040
'EF'                                                                     017050
LET STATUS.PRIORITY=0                                                    017060
FILE AMIS IN XX.BUFFER.QUEUE                                             017070
```

```
SCHEDULE A CPU.REQUEST IN 5.0*MIL.SEC.DAYS DAYS                    017080
SCHEDULE A IS.BUFFER.EMPTY NOW                                     017090
RETURN                                                             017100
'REESTABLISH'                                                      017110
LET XX.CPU.SERVICE.TIME(AMIS) = XX..CPU.SERVICE.TIME(AMIS)         017120
CREATE A IO.DELAY                                                  017130
LET NAME(IO.DELAY) = AMIS                                          017140
LET STATUS.PRIORITY=0                                              017150
IF FLAG = 1                                                        017160
                LIST JOBNUM(AMIS)                                  017170
                LIST IO.SERVICE.TIME(AMIS)                         017180
ELSE                                                               017190
ALWAYS                                                             017200
LET GAIN=GAIN + IO.SERVICE.TIME(AMIS)                              017210
SCHEDULE THE IO.DELAY IN IO.SERVICE.TIME(AMIS) DAYS                017220
SCHEDULE A CPU.REQUEST NOW                                         017230
RETURN                                                             017240
'FILE'                                                             017250
LET STATUS.PRIORITY=0                                              017260
FILE AMIS IN CPU..REQUEST.QUEUE                                    017270
SCHEDULE A CPU.REQUEST NOW                                         017280
RETURN                                                             017290
END                                                                017300
                                                                   017310
                                                                   017400
EVENT LOAD.DELAY                                                   017410
REMOVE FIRST AMIS FROM LOAD.QUEUE                                  017420
LET STATUS.PRIORITY=0                                              017430
LET XXTIME.E.CPU.QUEUE(AMIS)=TIME.V                                017440
FILE AMIS IN CPU..REQUEST.QUEUE                                    017450
LET CPU.SF = 0                                                     017460
LET LOAD.F= 0                                                      017470
LET F.LOADER(AMIS) = 0                                             017480
SCHEDULE A CPU.REQUEST NOW                                         017490
RETURN                                                             017500
END                                                                017510
```

```
EVENT IO.DELAY  SAVING THE EVENT NOTICE                                  017590
LET AMIS= NAME(IO.DELAY)                                                 017600
DESTROY IO.DELAY                                                         017610
LET XXTIME.E.CPU.QUEUE(AMIS) = TIME.V                                    017620
FILE AMIS IN CPU..REQUEST.QUEUE                                          017630
SCHEDULE A CPU.REQUEST NOW                                               017640
RETURN                                                                   017650
END                                                                      017660
EVENT RESPONSE SAVING THE EVENT NOTICE                                   017770
**************************************************************************017780
**************************************************************************017790
LET AMIS = RNAME(RESPONSE)                                               017800
DESTROY RESPONSE                                                         017810
LET XXTIME.E.CPU.QUEUE(AMIS)=TIME.V                                      017820
FILE AMIS IN CPU..REQUEST.QUEUE                                          017830
SCHEDULE A CPU.REQUEST NOW                                               017840
RETURN                                                                   017850
END                                                                      017860
                                                                         017870
EVENT IS.BUFFER.EMPTY                                                    017950
**************************************************************************017960
**     B.SF IS THE BUFFER STATUS FLAG                                   *017970
**          0 MEANS BUFFER EMPTY                                        *017980
**          1 MEANS BUFFER NOT EMPTY                                    *017990
**************************************************************************018000
CALL BUFFER YIELDING B.SF                                                018010
REMOVE FIRST AMIS FROM XX.BUFFER.QUEUE                                   018020
IF B.SF = 0                                                              018030
FILE AMIS IN ACM.RELEASE.QUEUE                                           018040
SCHEDULE A CM.RELEASE NOW                                                018050
ELSE                                                                     018060
FILE AMIS IN XX.BUFFER.QUEUE                                             018070
SCHEDULE A IS.BUFFER.EMPTY IN (1.0/1000.0)*SEC.DAYS DAYS                 018080
ALWAYS                                                                   018090
RETURN                                                                   018100
END                                                                      018110
                                                                         018120
```

```
EVENT CM.RELEASE                                                        018200
**********************************************************************  018210
**      START MAIN.MEMORY IS THE BEGINNING CORE LOCATION              *  018220
**      FINISH.MAIN.MEMORY IS THE LAST CORE LOCATION                  *  018230
**      MAIN.MEMORY IS THE MAIN MEMORY ARRAY                          *  018240
**********************************************************************  018250
REMOVE FIRST AMIS FROM ACM.RELEASE.QUEUE                                018260
LET SMM = START.MEMORY(AMIS)                                            018270
LET FMM = END.MEMORY(AMIS)                                              018280
FOR I = SMM TO FMM, DO                                                  018290
LET MAIN.MEMORY(I) = 0                                                  018300
LOOP                                                                    018310
FILE AMIS IN AN.OUTPUT.QUEUE                                            018320
SCHEDULE A OUTPUT NOW                                                   018330
RETURN                                                                  018340
END                                                                     018350
                                                                        018450
EVENT OUTPUT                                                            018460
**********************************************************************  018470
**      CHANNEL = 0 MEANS CHANNEL AVAILABLE                           *  018480
**      CHANNEL = 1 MEANS CHANNEL BUSY                                *  018490
**********************************************************************  018500
REMOVE FIRST AMIS FROM AN.OUTPUT.QUEUE                                  018510
CALL IO GIVEN AMIS YIELDING CHANNEL                                     018520
IF CHANNEL = 1                                                          018530
FILE AMIS IN HH.OUTPUT.QUEUE                                            018540
SCHEDULE A XX.OUTPUT IN 1.0 * MIL.SEC.DAYS DAYS                         018550
ELSE                                                                    018560
FILE AMIS IN AN.OUTPUT.QUEUE                                            018570
SCHEDULE A AN.DISK.RELEASE NOW                                          018580
ALWAYS                                                                  018590
RETURN                                                                  018600
END                                                                     018610
                                                                        018620
```

```
EVENT XX.OUTPUT                                                          018720
**********************************************************************   018730
**********************************************************************   018740
REMOVE FIRST AMIS FROM HH.OUTPUT.QUEUE                                   018750
FILE AMIS IN AN.OUTPUT.QUEUE                                             018760
SCHEDULE A OUTPUT NOW                                                    018770
RETURN                                                                   018780
END                                                                      018790
                                                                         018800
ROUTINE TO GIVEN AMIS YIELDING CHANNEL                                   018810
**********************************************************************   018820
**      CHANNEL = 0 MEANS CHANNEL AVAILABLE                           *   018830
**      CHANNEL = 1 MEANS CHANNEL BUSY                                *   018840
**********************************************************************   018850
IF JOB.TYPE(AMIS) EQ 2                                                   018860
     LET CH=RANDI.F(1,10,6)                                              018870
     IF CH LE 2                                                          018880
     LET CHANNEL=1                                                       018890
     ELSE                                                                019000
     LET CHANNEL=0                                                       019010
     ALWAYS                                                              019020
ELSE                                                                     019030
LET CH = RANDI.F(1,5,8)                                                  019040
  IF CH LE 2                                                             019050
  LET CHANNEL = 1                                                        019060
  ELSE                                                                   019070
  LET CHANNEL=0                                                          019080
  ALWAYS                                                                 019090
ALWAYS                                                                   019100
RETURN                                                                   019110
END                                                                      019120
                                                                         019220
EVENT AN.DISK.RELEASE                                                    019230
DEFINE V,W,AND Y AS REAL VARIABLES                                       019240
LET POUTFLAG=0
```

```
**********************************************************************
**     DISK CONTROLLER ROUTINE WILL RELEASE THE APPROPRIATE DISKS    *
**     NSUI IS NUMBER OF SIMULTANEOUS INTERACTIVE USERS              *
**     NSUB IS NUMBER OF SIMULTANEOUS BATCH USERS                    *
**********************************************************************
REMOVE FIRST AMIS FROM AN.OUTPUT.QUEUE
LET JOB.EXIT.TIME(AMIS) = TIME.V
LET TURNAROUND(AMIS) = TIME.V-ARRIVAL.TIME(AMIS)
LET STATUS.PRIORITY = 0
IF JOB.TYPE(AMIS) = 1
PRINT 1 LINE WITH JOBNUM(AMIS) AND TIME.V THUS
BATCH JOB# ***. MADE IT OUT AT TIME   *.********  ------------------
  LET NBJ = NBJ-1
  IF WARM.UP.FLAG=0
  LET CU=(TCPUT/TIME.V)*100.0
  LIST CU
  GO TO 'WARMING.UP.PERIOD'
  ELSE
     LET B.TURNAROUND.T = TURNAROUND(AMIS)
     LET B.CAPABILITY=TURNAROUND(AMIS)/CP.TIME(AMIS)
     LET V = TIME.V-ARRIVAL.TIME(AMIS)
     LET W=(CP.TIME(AMIS)+IO.TIME(AMIS))
     LET Y=CP.TIME(AMIS)
     LET ETME=V/W
     LET EX.DELAY.FACTOR=V/Y
ELSE
PRINT 1 LINE WITH JOBNUM(AMIS) AND TIME.V THUS
INTERACTIVE JOB NUM ***. MADE IT OUT AT TIME *.********   ----------
  LET NIJ = NIJ-1
  IF WARM.UP.FLAG=0
  LET CU=(TCPUT/TIME.V)*100.0
  LIST CU
  GO TO 'WARMING.UP.PERIOD'
  ELSE
     LET I.TURNAROUND.T = TURNAROUND(AMIS)
     LET I.CAPABILITY= TURNAROUND(AMIS)/CP.TIME(AMIS)
```

```
      LET V = TIME.V-ARRIVAL.TIME(AMIS)                                  019610
      LET W=(CP.TIME(AMIS)+IO.TIME(AMIS))                                019620
      LET Y=CP.TIME(AMIS)                                                019630
      LET I.ETME = V/W                                                   019640
      LET I.EX.DELAY.FACTOR=V/Y                                          019650
    ALWAYS                                                               019660
                    IF FLAG=1                                            019670
                    LIST STATUS.PRIORITY                                 019680
                    LIST TIME.V                                          019690
                    LIST PRIORITY(AMIS)                                  019700
                    ELSE                                                 019710
                    ALWAYS                                               019720
LET N.JOBS.PROCESSED = N.JOBS.PROCESSED+1                                019730
LET GAIN.FACTOR=GAIN/(TIME.V-W.UP.TIME)                                  019740
LET CURRENT.UTILIZATION =(T.CPU.TIME/(TIME.V-W.UP.TIME))*100.0           019750
LIST CURRENT.UTILIZATION                                                 019760
'WARMING.UP.PERIOD'                                                      019770
DESTROY THE JOB CALLED AMIS                                              019780
  IF JN GT 200                                                           019790
  SCHEDULE A CEASE.SIM NOW                                               019800
  ELSE                                                                   019810
  ALWAYS                                                                 019820
RETURN                                                                   019830
END                                                                      019840
                                                                         019850
ROUTINE BUFFER YIELDING B.SF                                             019960
**************************************************************************019960
**          B.SF IS THE BUFFER STATUS FLAG                              *019970
**          0 MEANS BUFFER EMPTY                                        *019980
**          1 MEANS BUFFER NOT EMPTY                                    *019990
**************************************************************************020000
LET B.SF=RANDI.F(1,10,4)                                                 020010
     IF B.SF LT 4                                                        020020
     LET B.SF=1                                                          020030
     ELSE                                                                020040
     LET B.SF=0                                                          020050
     ALWAYS                                                              020060
RETURN                                                                   020070
END                                                                      020080
```

```
EVENT CEASE.SIM                                                          020190
PRINT 1 LINE THUS                                                        020200
ENTERED CEASE.SIM                                                        020210
PRINT 1 LINE WITH TIME.V THUS                                            020220
  SIMULATION TIME IS   ****.*****                                        020230
  'RECHECK'                                                              020240
  FOR EACH AMIS IN CPU..REQUEST.QUEUE, WITH                              020250
  JOB.TYPE(AMIS) GT 0,                                                   020260
  FIND THE FIRST CASE                                                    020270
  IF NONE GO TO 'CONTINUE'                                               020280
  ELSE                                                                   020290
  REMOVE AMIS FROM CPU..REQUEST.QUEUE                                    020300
     IF JOB.TYPE(AMIS) EQ 1                                              020310
     LET B.BT=BLOCKED.TIME(AMIS)                                         020320
     ELSE                                                                020330
     LET I.BT=BLOCKED.TIME(AMIS)                                         020340
     ALWAYS                                                              020350
     DESTROY THE JOB CALLED AMIS                                         020360
     GO TO 'RECHECK'                                                     020370
'CONTINUE'                                                               020380
LIST JN                                                                  020390
LIST NJNPLO                                                              020400
LIST NIJ                                                                 020410
LIST NBJ                                                                 020420
PRINT 3 LINES THUS                                                       020430
           I N T E R A C T I V E   S T A T I S T I C S                   020440
                                                                         020450
                                                                         020460
PRINT 1 LINE THUS                                                        020470
                    NUMBER OF INTERACTIVE USERS                          020480
PRINT 1 LINE THUS                                                        020490
MEAN VALUE                  MAXIMUM NUMBER                 MINIMUM NUMBER020500
PRINT 2 LINES WITH ANSUI, BNSUI, AND CNSUI THUS                          020510
***.                        ***.                            ***.         020520
                                                                         020530
```

```
PRINT 2 LINES THUS                                                      020540
                INTERACTIVE TURNAROUND TIME                             020550
MEAN                        VARIANCE                        STD.DEV     020560
PRINT 2 LINES WITH AI.TURNAROUND.T, BI.TURNAROUND.T, AND CI.TURNAROUND.T THUS 020570
*.******                   *.******                        *.******    020580
                                                                        020590
PRINT 2 LINES THUS                                                      020600
                    INTERACTIVE CPU TIME                                020610
MEAN                        VARIANCE                        STD.DEV     020620
PRINT 2 LINES WITH AI.CPU.TIME, BI.CPU.TIME, AND CI.CPU.TIME THUS       020630
*.******                   *.******                        *.******    020640
                                                                        020650
PRINT 2 LINES THUS                                                      020660
                  INTERACTIVE CAPABILITY                                020670
MEAN                MEAN SQUARE                 VARIANCE        STD.DEV 020680
PRINT 2 LINES WITH AA,BB,CC,AND DD THUS                                 020690
****.               ****.                       ****.           ****.   020700
                                                                        020710
PRINT 2 LINES THUS                                                      020720
                  INTERACTIVE CPU BLOCKED TIME                          020730
MEAN                MEAN SQUARE                 VARIANCE        STD.DEV 020740
PRINT 2 LINES WITH AAA, BBB, CCC, AND DDD THUS                          020750
*.******            *.******                    *.******        *.******020760
                                                                        020770
PRINT 2 LINES THUS                                                      020780
              INTERACTIVE MEMORY.REQUESTS IN K                          020790
MEAN                MEAN SQUARE                 VARIANCE        STD.DEV 020800
PRINT 2 LINES WITH II, JJ, KK, AND LL THUS                              020810
****.               ****.                       ****.           ****.   020820
                                                                        020830
PRINT 2 LINES THUS                                                      020840
               INTERACTIVE USER RESPONSE TIME                           020850
MEAN                MEAN SQUARE                 VARIANCE        STD.DEV 020860
PRINT 2 LINES WITH QQ,RR,SS, AND TT THUS                                020870
****.               ****.                       ****.           ****.   020880
                                                                        020890
```

```
PRINT 4 LINES WITH I.E, I.EA, I.EB AND I.EC THUS                          020900
               ELASPSED TIME MULTIPROGRAMMING FACTOR                      020910
MEAN              VARIANCE              STD.DEV              MEAN.SQUARE   020920
***.*             ***.*                 ***.*                ***.*         020930
                                                                           020940
PRINT 2 LINES WITH IIO THUS                                                020950
     MEAN INTERACTIVE IO TIME                                              020960
     *.********                                                            020970
PRINT 9 LINES WITH I.EDEA, I.EDEB, I.EDEC AND I.EDED THUS                  020980
                              EXTERNAL DELAY FACTOR                        020990
MEAN              VARIANCE              STD DEV              MEAN SQUARE   021000
***.*             ***.*                 ***.*                ***.*         021010
                                                                           021020
                     B A T C H   S T A T I S T I C S                       021030
                                                                           021040
                                                                           021050
                        NUMBER OF BATCH JOBS                               021060
MEAN                  MAXIMUM NUMBER                  MINIMUM NUMBER       021070
PRINT 2 LINES WITH ONSUB, ENSUB AND FNSUB THUS                             021080
***.                  ***.                            ***.                 021090
                                                                           021100
PRINT 2 LINES THUS                                                         021110
                           BATCH TURNAROUND                                021120
MEAN                          VARIANCE                          STD.DEV    021130
PRINT 2 LINES WITH AB.TURNAROUND.T, BB.TURNAROUND.T, AND CB.TURNAROUND.T THUS 021140
*.******                     *.******                          *.******   021150
                                                                           021160
PPINT 2 LINES THUS                                                         021170
                        BATCH CPU TIME                                     021180
MEAN                          VARIANCE                          STD.DEV    021190
PRINT 2 LINES WITH AB.CPU.TIME, BB.CPU.TIME, AND CB.CPU.TIME THUS          021200
*.******                     *.******                          *.******   021210
                                                                           021220
PRINT 2 LINES THUS                                                         021230
                           BATCH CAPABILITY                                021240
MEAN              MEAN SQUARE              VARIANCE              STD DEV    021250
PRINT 2 LINES WITH EE, FFFF, GG, AND HH THUS                               021260
****.*            ****.*                   *****.*               ****.*     021270
                                                                           021280
```

```
PRINT 2 LINES THUS
                        BATCH CPU BLOCKED TIME
MEAN                MEAN SQUARE              VARIANCE              STD DEV
PRINT 2 LINES WITH EEE,FFF, GGG, AND HHH THUS
*.******            *.******                 *.******              *.******

PRINT 2 LINES THUS
                       BATCH MEMORY REQUEST IN K
MEAN                MEAN SQUARE              VARIANCE              STD DEV
PRINT 2 LINES WITH MM,  NN, OO, AND PP THUS
***.                ***.                     ***.                  ***.

PRINT 5 LINES WITH UU,VV,WW AND XX THUS
              ELASPSED TIME MULTIPROGRAMMING FACTOR
MEAN                VARIANCE              STD.DEV              MEAN.SQUARE
***.*               ***.*                 ***.*                ***.*

                  EXTERNAL DELAY FACTOR
PRINT 4 LINES WITH UUU,VVV,WWW, AND XXX THUS
MEAN                VARIANCE              STD DEV              MEAN SQUARE
***.*               ***.*                 ***.*                ***.*


PRINT 2 LINES WITH BIO THUS
     MEAN BATCH IO TIME
     *.********
PRINT 2 LINES WITH TOTAL.JOB.CPU.TIME AND TOT.JOB.IO.TIME THUS
     TOTAL JOB CPU TIME       TOTAL JOB IO TIME
     *.********               *.********
LIST N.JOBS.PROCESSED
PRINT 4 LINES WITH GAIN.FACTOR THUS


                              GAIN.FACTOR
                               ***.****
LET CURRENT.UTILIZATION =(T.CPU.TIME/(TIME.V - W.UP.TIME))*100.0
LIST CURRENT.UTILIZATION
PRINT 1 LINE THUS
EXIT CEASE.SIM
STOP
END
```

VITA

Alfred H. Linder, III was born on 14 June 1948 in San Diego, California. He graduated from Flagstaff High School in Flagstaff, Arizona in 1966 and enlisted in the United States Navy in June of 1966. After three years in the Navy, as an electronic technician, he joined the Navy Reserve while working for the Sante Fe Railroad and then the Mountain Bell Telephone Company. He attended Northern Arizona University where he received, in 1976, a Bachelor of Science in Mathematics and a commission in the United States Air Force. He entered the AFIT residence school in August 1976 and received the degree of Master of Science in Computer Science in Dec 1978.

Permanent Address: 418 Sunset
Winslow, Arizona 86047

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GCS/MA/78-1	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Simulation of the Acquisition Management Information System		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Alfred H. Linder III		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN)		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Computer System Division AFSC/PPQA Wright Patterson AFB, Ohio		12. REPORT DATE March, 1978 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE
IAW AFR 190-17.

JERRAL C. GUESS, Capt, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Simulation modeling
Data Base Management System
Computer Performance Measures
Computer Throughput Performance
Batch-Interactive-Mix Analysis
Simscript II.5

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A throughput and batch-interactive-mix analysis was performed on a IBM 370/155 computer. Variables of interest in conducting this simulation analysis include CPU utilization and gain factor. Different computer performance measures are discussed in the development of this SIMSCRIPT II.5 computer program.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)